Genetic Programming and Evolvable Machines (2018) 19:391–419
https://doi.org/10.1007/s10710-018-9330-7

Data exploration in evolutionary reconstruction of PET images

Cameron C. Gray[1] · **Shatha F. Al-Maliki**[1,2] · **Franck P. Vidal**[1]

Received: 11 December 2017 / Revised: 5 July 2018 / Published online: 1 August 2018

Abstract

This work is based on a cooperative co-evolution algorithm called 'Fly Algorithm', which is an evolutionary algorithm (EA) where individuals are called 'flies'. It is a specific case of the 'Parisian Approach' where the solution of an optimisation problem is a set of individuals (e.g. the whole population) instead of a single individual (the best one) as in typical EAs. The optimisation problem considered here is tomography reconstruction in positron emission tomography (PET). It estimates the concentration of a radioactive substance (called a radiotracer) within the body. Tomography, in this context, is considered as a difficult ill-posed inverse problem. The Fly Algorithm aims at optimising the position of 3-D points that mimic the radiotracer. At the end of the optimisation process, the fly population is extracted as it corresponds to an estimate of the radioactive concentration. During the optimisation loop a lot of data is generated by the algorithm, such as image metrics, duration, and internal states. This data is recorded in a log file that can be post-processed and visualised. We propose using information visualisation and user interaction techniques to explore the algorithm's internal data. Our aim is to better understand what happens during the evolutionary loop. Using an example, we demonstrate that it is possible to interactively discover when an early termination could be triggered. It is implemented in a new stopping criterion. It is tested on two other examples on which it leads to a 60% reduction of the number of iterations without any loss of accuracy.

Keywords Fly Algorithm · Tomography reconstruction · Information visualisation · Data exploration · Artificial evolution · Parisian evolution

Abbreviations

EA	Evolutionary algorithm
PET	Positron emission tomography
ET	Emission tomography

✉ Franck P. Vidal
f.vidal@bangor.ac.uk

Extended author information available on the last page of the article

InfoVis	Information visualisation
CSV	Comma-Separated Values
CCEA	Cooperative co-evolution algorithm
CT	Computed tomography
SPECT	Single-photon emission computed tomography
SNR	Signal-to-noise ratio
MRI	Magnetic resonance imaging
keV	Kiloelectron volt
LOR	Line of response
MLEM	Maximum-Likelihood Expectation-Maximization
EM	Expectation-Maximization
OSEM	Ordered Subset Expectation-Maximization
voxel	Volume element
SVG	Structured Vector Graphics
MAE	Mean absolute error
MSE	Mean squared error
RMSE	Root mean squared error
ZNCC	Zero-normalised cross-correlation
PSNR	Peak signal-to-noise ratio
SSIM	Structural similarity
DSSIM	Structural dissimilarity
TV	Total variation

1 Introduction

This research is related to the use of evolutionary computing in nuclear medicine, more particularly positron emission tomography (PET) reconstruction. In this paper, we investigate the use of information visualisation (InfoVis) and data exploration to understand some of the behaviours of an evolutionary algorithm (EA). In particular, we want to assess if the algorithm could have been stopped earlier to get a reasonable solution instead of waiting until the algorithm ends and using the final solution as the problem answer.

The combination of visualisation and evolutionary computing is still a relatively overlooked field. Two different approaches can be distinguished:

- visualisation to understand an evolutionary algorithm [30, 38, 52, 53], and
- interactive artificial evolution to improve the visualisation [9, 24, 36].

First attempts were reported at the end of the 90s. Early visualisations were using relatively basic techniques that mostly relied on plotting with limited or no interactivity. During the evolutionary PET reconstruction, multiple time series are recorded hundreds of thousands of times. Comparing these time series by hand using typical scatterplots and line charts with no interactivity is not practically feasible:

- The order of magnitude of each time series is different. They would need to be independently normalised before plotting.
- Trial and error would be needed to choose the axis of interest because it is not necessarily straightforward to do so without a deep *a priori* understanding of the data.
- Adjusting the data range visualised in the scatterplots would also need to be performed with trial and error.
- Displaying selected images would need to be done manually.

The use of Parallel Coordinate Plots is very popular to visualise high-dimensional geometry and analyse multivariate data [28], which is the type of data considered here. Interactivity using the brushing technique [37] makes it feasible to easily explore parts of this high-dimensional space and visually analyse this complex multivariate dataset. This is the approach we adopted here to develop an integrated visualisation framework dedicated to our evolutionary PET reconstruction algorithm.

The emergence of information visualisation and data analytics is opening new doors for its use in the evolutionary computing domain. We adopt the first approach to analyse the evolutionary process of our image reconstruction algorithm for tomography in nuclear medicine. In typical evolutionary algorithms, the best individual of the final population is the solution of the optimisation problem. Our algorithm relies on the Parisian approach where the solution to the problem is a group of individuals, e.g. the whole population or a subset of the population. The population size progressively increases to improve the resolution of the output image. The algorithm is launched with input parameters such as the initial number of individuals, the final number of individuals, the probability of operators, etc., the final solution is extracted at the end of the optimisation process then converted into a problem-specific answer. A lot of the data is generated during the evolution process, in particular data based on error metrics and correlation measurements. Traditionally all this data is discarded at the end of the evolutionary process, as only the final population is considered. Our hypothesis is that intermediate populations and internal data should not be systematically discarded as they can be reviewed offline. They can be used to analyse the performance of the population over time. When using stagnation as the stopping criterion, the final population is not necessarily the best one due to oscillations around the minimal fitness value. In such a case, past generations will have to be accessible. Also, reaching the targeted number of individuals may not be necessary if the reconstructed image stops improving. Offline analysis of intermediate results makes it possible to look at quality metrics other than the fitness value, e.g. smoothness of the reconstructed image. Our initial goal is to extract the best possible solution in terms of fitness function and smoothness of the reconstructed image. The aim is to identify the smallest population that could be used as the solution instead of the final population to reduce the computing time as much as possible without compromising the quality of the reconstructed PET image. In the case study considered below 20 measurements were repeated thousands of times at regular intervals during the evolutionary process. It corresponds to a dataset that includes several hundreds of thousands of samples. Multivariate analysis can be

used to highlight relationships and correlations in the dataset [7]. Analytic tasks that are involved include value retrieving, filtering, extremum identification, and range determination [6]. For this purpose, we use interactive parallel coordinate and scatterplots to visually analyse this dataset.

Our contribution demonstrates how simple interactive visualisation techniques such as Parallel Coordinate Plots, scatterplot and image display can be used to analyse complex datasets generated using the temporal internal data of the evolutionary algorithm. The paper illustrates how it can be used to analyse the behaviour of the algorithm over time. This task would be extremely difficult without interactive visualisation. Well-designed user interaction and effective visualisation make it relatively easy. Our approach can be generalised to improve the performance of other special purpose evolutionary algorithms. We have re-implemented our evolutionary reconstruction algorithm to output a large Comma-Separated Values (CSV) log file with time series of image metrics (including error distances, correlation measurements, and smoothness) as well as internal states of the algorithms (e.g. iteration number, population size, and average probability of genetic operators) and store intermediate results (e.g. reconstructed images). A simple, but yet effective, visualisation framework has been purposely developed to explore data embedded in the log file and display the intermediate results based upon user interactions. It is used to assess the behaviour of the evolution process over time. The relationship between different properties of the reconstruction can also be examined. Using a case study, it helped us to ascertain that allocating more computation time to the reconstruction algorithm did not lead to a significant improvement in accuracy. We propose an alternative early stopping criterion that looks at both the global fitness of the population and the smoothness of the reconstructed image over the last 500 iterations. It is tested multiple times using three test cases with and without this new stopping criterion.

Section 2 is a brief introduction to the application considered here: nuclear medicine and PET imaging. It provides an insight into the main principles of nuclear medicine and how they can be used in imaging to provide a 3-D map of radioactive concentration through the patient's body. The evolutionary framework, Parisian Evolution, used in this application is described in Sect. 3. In evolutionary computing, the answer to the optimisation problem is a single individual, the one with the best fitness. Parisian Evolution is a class of Cooperative Co-evolution Algorithms (CCEAs) where the answer to the optimisation problem is a group of individuals (e.g. the whole population). Section 4 presents the Fly Algorithm. It is an example of Parisian Evolution dedicated to imaging problems such as computer stereo vision and tomography reconstruction. Its application to tomography reconstruction is described in Sect. 5. The next section develops the information visualisation techniques that we used to explore the internal data generated by successive iterations of the Fly Algorithm. Section 7 discusses how the visualisation is exploited to help us select the 'best' reconstructed image. It is also used to design a new early stopping criterion. It is followed by a conclusion that summarises our contributions and it provides ideas for further work. For clarity, a glossary of (mathematical) terms specific to this application in nuclear imaging, and a list of acronyms are provided at the end of the article.

2 Emission tomography in nuclear medicine

In nuclear medicine, a radiopharmaceutical (i.e. radioactive substance) is administered to the patient. The radioisotope is fixed on a given molecule that is going to be absorbed by the body in relation to a targeted physiological process, such as tumour growth, bone fracture, or reduced blood flow in the heart. Imaging in this context is a type of *molecular and functional* imaging. In other words, a physiological function is targeted by the radioactive molecule. In Oncology, the molecule will be fixed by tumours because of the growth of cancerous cells. This is why tumours are highlighted in Fig. 1. The radioactive concentration is a lot higher in tumours than healthy tissues, which leads to more emission from the tumours. For this reason tomography in nuclear medicine is called emission tomography (ET): The source of radiation is within the patient. There are two main techniques: single-photon emission computed tomography (SPECT) and PET. They both produce a stack of 2-D cross-sections through the human body, which corresponds to a 3-D map of the radioactive concentration within the patient (see Fig. 1b). We focus in this paper on PET as it is now the main technique in nuclear medicine imaging.

Figure 2 shows the principle of the PET data acquisition chain. Images produced in ET have a relatively low resolution (typically 128×128 pixels) and signal-to-noise ratio (SNR). Well-known tomography techniques used in radiology departments, such as computed tomography (CT) and magnetic resonance imaging (MRI) generate images with a much higher resolution (typically 512×512 pixels) and SNR (see Fig. 1a). They are used to visualise anatomical structures. Modern medical scanners now combine PET with either CT or MRI to provide collocated physiological and anatomical image datasets (see Fig. 1c).

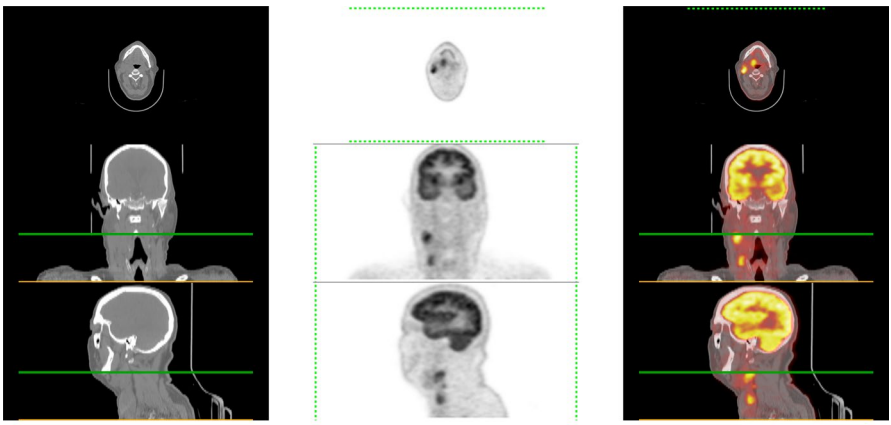

Fig. 1 PET-CT examination of a "head and neck" patient in Oncology. In this case tumours were not visible on the anatomic images from CT, but were on the physiological images from PET. Top row: axial plane; middle row: coronal plane; and bottom row: sagittal plane. **a** CT data, **b** PET data, **c** PET-CT data. *Source*: the Cancer Imaging Archive (http://www.cancerimagingarchive.net/) [21]

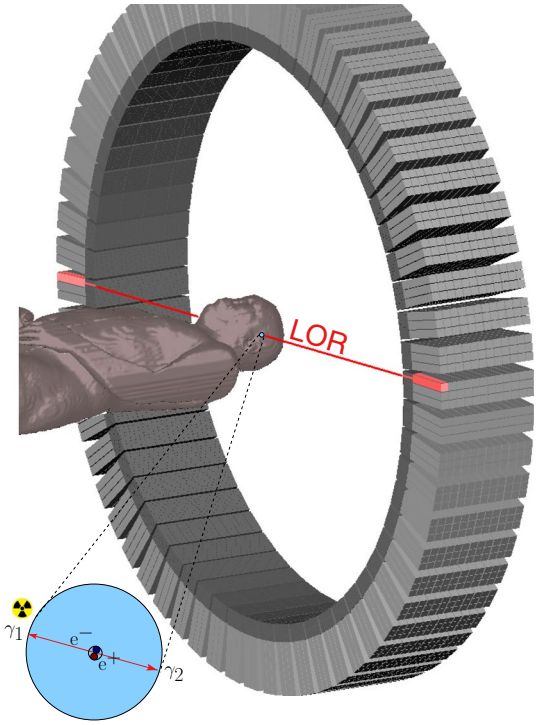

Fig. 2 PET data acquisition: 1 e^- combines with 1 e^+; it may results in an annihilation reaction, which generates 2 γ of 512 keV emitted at about 180°; the line joining the pair of detectors activated by this pair of γ is called LOR; the system records many LORs; this data is used by the reconstruction algorithm (possibly after conversion into a sinogram, see Fig. 3) (Color figure online)

To obtain such images by tomography, regardless of the modality, a reconstruction algorithm is performed after the data acquisition. It aims to generate the stack of 2-D images from the original projection data. It is therefore important to understand the data acquisition process in PET and what the projection data might be. In PET, the positron (often shortened as e^+ or β^+) is the type of ionising radiation that is used. When a positron collides with an electron (e^-), an annihilation reaction may occur. In this case, two photons (γ) are emitted at almost 180° of each other with a kinetic energy of 512 kiloelectron volts (keV). Note that photons are the elementary particle of light. Pairs of annihilation photons are detected in coincidence, i.e. at almost the same time, by a dedicated scanner. The line joining the two detectors (see red parallelepipeds in Fig. 2) that caught the photons of the same pair is called line of response (LOR) (see red line in Fig. 2). Each detector of the PET scanner has a unique identifier. All the pairs of detectors corresponding to the LORs are recorded by the system. However, the exact locations of the annihilation reaction are unknown. PET reconstruction aims at producing the 3-D volume of radioactive concentration from the recorded LORs. Dedicated algorithms can be used to exploit LOR data directly, this is called *list-mode reconstruction*. LOR data can also be converted into sinograms as this is a common data representation [23] that stores a set of 1-D projections at successive angles in a 2-D image (see Fig. 9). They can be used with conventional tomography reconstruction algorithms. To convert a LOR into a point into the sinogram, the angle between the LOR and the horizontal axis is computed (see α in Fig. 3). The shortest distance between the LOR and the origin of

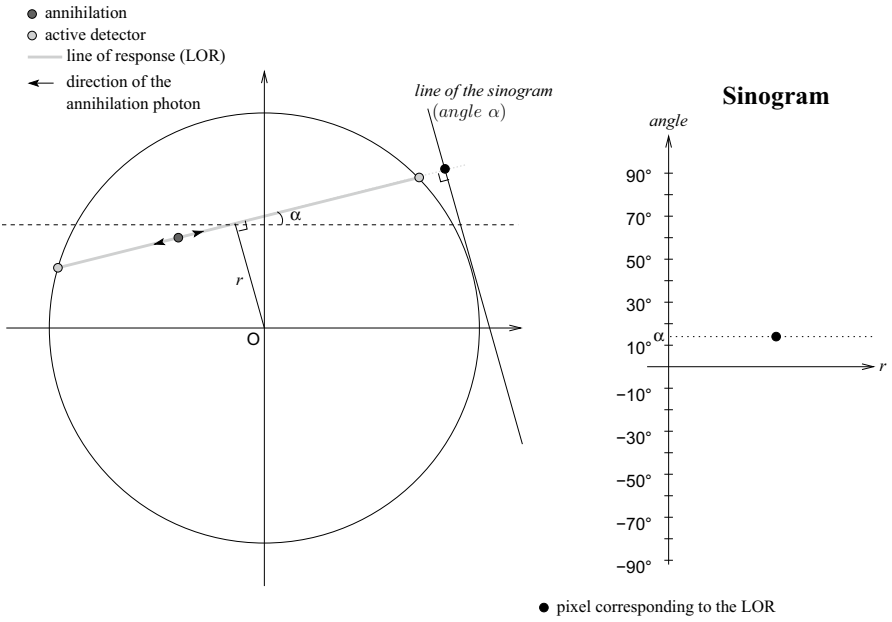

Fig. 3 Conversion of LOR data into a sinogram

the system is also computed (see r in Fig. 3). The intensity of pixel(r, α) in the sinogram is derived from the number of corresponding LOR events detected by the PET scanner. Note that we will use this format in this paper, although we demonstrated in [45, 46] that LOR data can be used within our algorithm.

Tomography reconstruction is an ill-posed inverse problem due to missing data and photonic noise (Poisson noise) in the measured photon count. Noise is actually a major concern in ET. Statistical iterative reconstruction takes into account Poisson noise in the measured photon count. This is why Maximum-Likelihood Expectation-Maximization (MLEM) and its derivatives, such as Ordered Subset Expectation-Maximization (OSEM) are the main methods used in nuclear medicine [27, 39, 43]. One of the main issues with MLEM-based algorithms is the difficulty to choose a good stopping criterion [13] as they rely on a non-converging method. When the number of iterations increases the reconstruction offers a better resolution but becomes noisy, which is not the case with our evolutionary reconstruction framework.

The reconstruction can be considered as an optimisation problem. Evolutionary computing is known to perform well, in general, when solving hard ill-posed problems, and in particular in medical imaging [15, 20, 49, 50]. It is a class of stochastic optimisation tool that relies on Darwin's principles to mimic complex natural behaviours [10]. A proof-of-concept of evolutionary reconstruction was initially developed with a relatively basic Fly Algorithm [45, 46]. The implementation supported list-mode reconstruction using LOR data as well as conventional reconstruction using sinograms. More advanced genetic operators (namely threshold selection, mitosis and dual mutation)

were proposed [47] and validated against commonly used genetic operators [48]. More sophisticated mutation operators were proposed and evaluated to speed up computations and to further improve the accuracy of the reconstructions [1]. The algorithm optimises the position of 3-D points that mimic pairs of annihilation photons. The output is, therefore, a set of points known as 'point cloud'. The extraction of the solution and how to convert this point cloud into volume elements (voxels) have already been addressed [5]. It relies on using implicit modelling and used the individuals' fitness as a confidence measurement to adjust their individual footprint in the final image. Results were comparable or better than those obtained with OSEM on the test cases used.

3 Parisian evolution

In Parisian Evolution, the population of individuals is considered as a society where the individuals collaborate toward a common goal. This is implemented using an EA that includes all the common genetic operators (e.g. mutation, cross-over, and selection). The main difference is in the fitness function landscape. In Parisian Evolution there are two levels of fitness function.

Local fitness function:	to assess the performance of a given individual. It is used during the selection process. For an individual, improving its local fitness means increasing its chances of breeding.
Global fitness function:	to assess the performance of the whole population. Improving (maximising or minimising depending on the problem considered) the global fitness is the goal of the population.

In addition, a diversity mechanism is required to avoid individuals gathering in only a few areas of the search space. Another difference between classical EAs and Parisian Evolution is in the extraction of the solution once the evolutionary loop terminates. In classical evolutionary approaches, the best individual corresponds to the solution and the rest of the population is discarded. Here, all the individuals (or individuals of a subgroup of the population) are collated to build the problem solution. The way the fitness functions are constructed and the way the solution is extracted, are problem-dependent.

Parisian Evolution has been successfully applied to various optimisation problems, such as text-mining [31], hand gesture recognition [29], complex interaction modelling in industrial agrifood processes [11, 12], and imaging problems [22] such as computer stereo vision [33] and tomography reconstruction [5].

4 Fly Algorithm and its applications

The Fly Algorithm is a good example of Parisian Evolution [17]. It was initially proposed in computer stereo vision to extract 3-D information from pairs of digital images. The algorithm is a fast evolutionary algorithm that can be used to detect the

location of obstacles [33, 34]. It is used in autonomous robot navigation to avoid collision with objects and walls.

The Fly Algorithm evolves a population of flies. Each fly is defined as a 3-D point with coordinates (x, y, z) in the solution space. A set of 3-D points is often called *point cloud* in the literature. Flies are projected to compute the local fitness function. This projection operator is problem-specific. In stereo vision applications, each fly is projected twice: once on the image taken from the left camera and once on the image taken from the right camera [35]. When a fly is located on the surface of an object, the pixel neighbourhood of its two projections will match; when a fly is not located on the surface of an object, the pixel neighbourhood of its two projections will be significantly different. The fitness function is designed to take advantage of this fact: The fitness of a fly measures the consistency between its two projections. The algorithm will optimise the 3-D position of the flies so that their projections on the left-hand side and right-hand side 2-D images are similar.

The Fly Algorithm is implemented as any other EA. It starts with a population of randomly generated individuals. They are the *parents*. Then, depending on the genetic operators (selection, mutation, new blood, etc.) that are applied to the parents, a population of new individuals, the *offspring*, is produced. Selection is used to pick up candidate parents for breeding. Mutation is used to randomly alter the genes of an individual. New blood corresponds to creating a randomly generated individual. This simple, but yet effective, operator preserves diversity in the population. Note that crossover is not generally used in the Fly Algorithm because if there are two good flies on different objects, creating a new one in between is likely to produce a bad fly. The fitness function determines the validity of a fly's position and it is calculated during the selection process. The new generation of offspring eventually becomes parents. The same operations are repeated until a stopping criterion is reached. This approach is called 'Generational Fly Algorithm' [42].

A Steady-State approach is also possible (see Fig. 4). Using this approach, a bad fly is selected at each iteration and replaced by a new one. The rationale is that the new one is likely to be better and there is no reason to delay using it [42]. We follow this approach for PET reconstruction.

A decade after the initial developments of the Fly Algorithm in robotics; it was adapted to SPECT reconstruction [18], then to PET [1, 5, 45–48]. It has also be used in filtering to generate artistic effects on images [2–4].

5 Evolutionary reconstruction in PET

The data acquisition in PET can be described as:

$$Y = Pf \tag{1}$$

where f is the radioactive concentration, which is unknown; Y the observations (known data as measured by the scanner); and P the system matrix or projection operator (it transforms the radioactive concentration into projections). For iterative reconstruction, scanner geometry, noise, etc. can be modelled in P. Tomography reconstruction corresponds to solving:

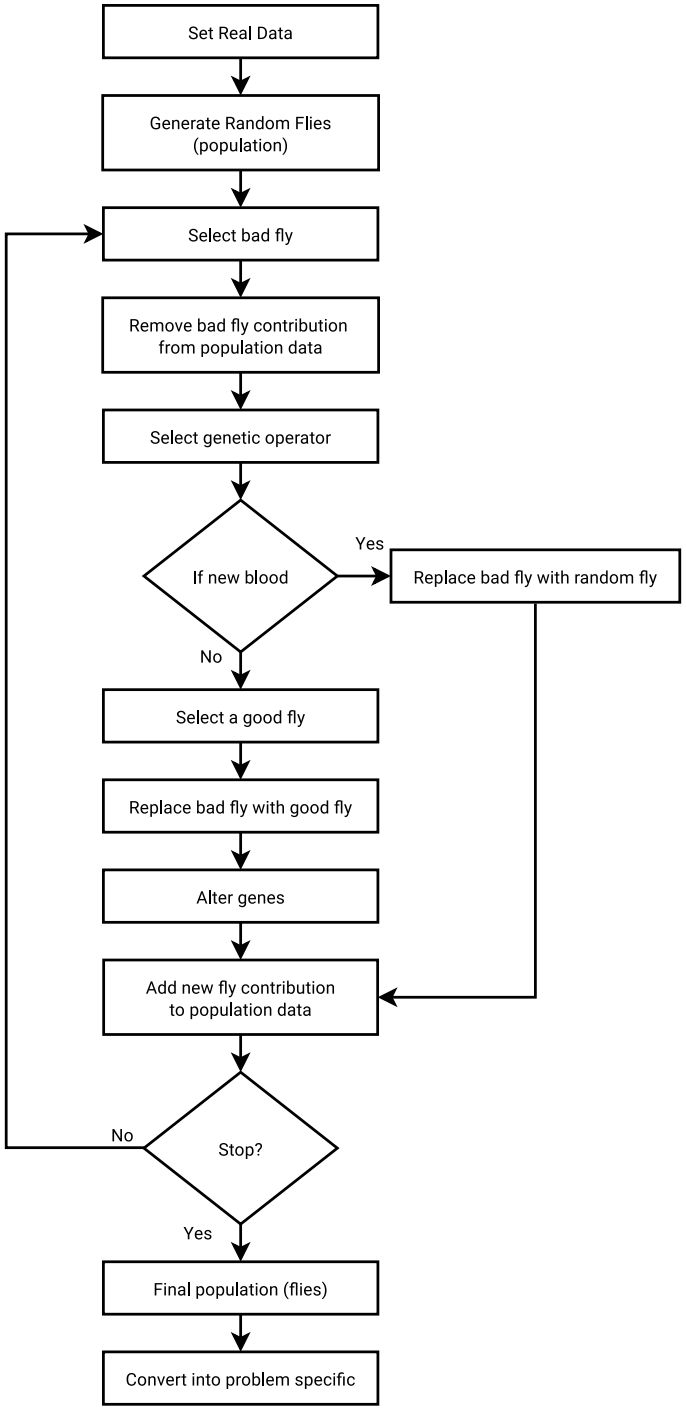

Fig. 4 Overall flowchart of a Fly Algorithm in steady-state

$$f = P^{-1}Y \tag{2}$$

where P^{-1} is the inverse transformation. This is why tomography reconstruction is an inverse problem. However directly applying the inverse transformation is not trivial due to noise in Y and missing data: tomography reconstruction is ill-posed. Optimisation can be used to produce an estimate $\hat{f}$ of f using Y so that:

$$\hat{f} = \arg\min \left\| Y - \hat{Y} \right\|_2^2 \tag{3}$$

where $\hat{Y}$ is the projection data corresponding to $\hat{f}$ ($\hat{Y} = P\hat{f}$); and $||Y - \hat{Y}||_2^2$ is the ℓ^2 -norm, also known as Euclidean distance, between Y and $\hat{Y}$:

$$\left\| Y - \hat{Y} \right\|_2^2 = \sqrt{\sum_{y=0}^{y<h} \sum_{x=0}^{x<w} \left[Y(x,y) - \hat{Y}(x,y) \right]^2} \tag{4}$$

Evolutionary reconstruction using the Fly Algorithm corresponds to the iterative paradigm (see Fig. 5). The initial guess is a population ($\hat{f}$) of flies randomly located within the object space. Projections ($\hat{Y}$) are computed from the population and are compared with the data (Y) from the medical scanner. To that effect, an error metric between the two images is measured (see Eq. 4), this is the global fitness. It is the numerical value that the optimisation algorithm will minimise. Errors are corrected using the application of genetic operators (mainly selection, mutation, new blood, and mitosis). The aim is to optimise the position of each fly so that the projection data of the whole population closely matches the one from the real radioactive concentration. The process is repeated until a stopping criterion is met. After convergence, the point cloud made by the flies is an estimate of the real radioactive concentration. The point cloud is then sampled to produce voxel data [5].

We employ a steady-state EA (i.e. evolution strategy of type $\mu/\rho + 1$) as in Fig. 4 where, at each iteration, a bad fly is selected for death and replaced using a genetic operator (mutation or new blood). To evaluate the performance of a single individual (Fly i), we use the *marginal fitness* ($F_m(i)$) [18]. It relies on the global fitness with the leave-one-out cross-validation principle:

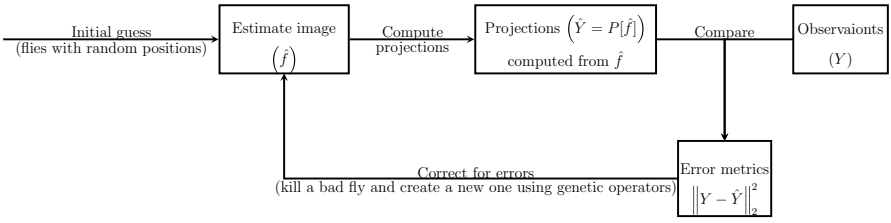

Fig. 5 Iterative reconstruction paradigm

$$F_m(i) = \left\| Y - \left(\hat{Y} \setminus \{i\} \right) \right\|_2^2 - \left\| Y - \hat{Y} \right\|_2^2 \tag{5}$$

where $\hat{Y} \setminus \{i\}$ is the estimated projections without the photons simulated by Fly i. The idea behind the leave-one-out cross-validation is to assess the error metric twice: once with Fly i in the population, and once without it. By comparing the two values (the subtraction in Eq. 5) we can determine if having Fly i is beneficial or not for the population. If F_m is positive, the error is smaller when the fly is included: the fly has a positive impact on the population's performance. It is a good fly, i.e. a good candidate for reproduction. If F_m is negative, the error is larger when the fly is included: the fly has a negative impact on the population's performance. It is a bad fly, i.e. a good candidate for death. F_m is, therefore, a measure maximised by the algorithm. We use this principle to define the 'threshold-selection' operator [47, 48]: to choose a fly to kill, find a fly with $F_m \leq 0$; and to choose a fly to reproduce, find a fly with $F_m > 0$. When the number of bad flies is low, the threshold-selection will struggle to find flies to kill. It provides a good stopping criterion.

Note that our implementation is multi-resolution and includes an extra loop that is not presented in Fig. 4. We start with a low number of flies (e.g. 25). When convergence is detected, each fly is duplicated to double the population size (see mitosis operator in [47, 48]). Each new fly is then mutated. Then the evolutionary process carries on until convergence is detected again. When the number of flies reaches a limit set by the user, and when convergence is detected, the reconstruction process ends. Note that details about our mutation operators are available in [1]. In the test cases presented below, we will use the new blood, basic mutation, and adaptive mutation operators. Also, note that the implementation is fully adaptive: operator probabilities are encoded by flies and undergo mutations.

Several stopping criteria can be used. Stagnation can be detected if the threshold selection operator struggles to find a bad fly several times in a row. The goal of the population is to minimise the global fitness as it is an error measurement. Stagnation can also be detected if the global fitness stops decreasing over a given number of iterations.

6 Data exploration

To assist in tuning the evolutionary algorithm, the complex interplay of each metric needs to be understood. Examining the raw data in numeric form is often error-prone and limited by the exact process employed by the researcher. There are other methods, such as writing bespoke analysis programs or the use of summary statistics in a spreadsheet application. These methods are then limited by capability of the tools, and results are still provided in text form which can be harder to reason with. The field of Visual Analytics provides another alternative; exploiting the visual processing and reasoning abilities of the human being [26]. Systems built for visual analytics can be expanded to use multiple views [41] of the same dataset highlighting deeper relationships and patterns. Therefore, we substitute the exact metric value

for a graphical and/or spatial surrogate. In this form, relation of metrics becomes an easier task requiring less mathematical and domain-specific knowledge.

The Fly implementation produces a multivariate output, which may or may not be interrelated. In order to achieve the goal of inferring those relationships, our design choices are limited to multivariate relationship techniques. The common options in this situation are Heatmaps, Marimekko Charts, Parallel Coordinate Plots, Radar Charts, and Venn diagrams [40]. As Heatmaps and Marimekko charts are limited in the number of variables they can display [51], and Venn diagrams become difficult to read beyond three variables; we must discount these options. Radar Charts are able to handle a larger number of variables, limited by the sweep angle between each axis. In theory without needing actual scale values, the chart could support 360 different axes; however, in practice the limit is substantially lower. An additional factor is that individuals (results in our case) are plotted over each other. Even with opacity effects it becomes more difficult to visually separate the individuals or extract patterns.

This requirements analysis leaves Parallel Coordinate Plots as the logical choice [54]. Parallel Coordinate Plots [28], first popularised in computerised form by Alfred Inselberg, visualise data in the form of multiple linked axes on one graph. The plot will still suffer from over-plotting where results share equal/similar values. These axes are scaled such that each domain is represented in the same length. These axes represent different measurements, or facets, of the objects in the dataset. Objects, known as instances, are plotted as a traditional straight line graph on these co-measurable axes. These plots are used to identify clusters [55] and identify properties of those clusters/subsets [8].

Visualisations are most effective when they not only show information but allow a user to answer their own questions by interacting with the data [32]. The tool allows the axes to be re-positioned and re-ordered to make any relationships more clear. Parallel Coordinate Plots most often deal with ranges rather than individuals. When selecting ranges of data, most tools implement the Brushing [37] technique. This allows the user to select multiple items in one stroke as if they were being painted with a brush. Our tool allows as many brushed ranges as there are axes, allowing users to precisely select items of interest, removing or fading unrelated data from the view. Off-the-shelf computer programs, such as Tableau [44] or Grapheur [14, 19], can be readily used to perform the visualisation of CSV files using Parallel Coordinate Plots and scatterplots. However, the customised task-specific interactions, highlighted later, may not be possible.

The proposed visualisation is produced using a browser-based library, D3.js [16], which produces Structured Vector Graphics (SVG) images. The library is written in JavaScript, with the visualisation code also written in JavaScript. D3 includes multiple methods for loading and processing data. This system processes the CSV log files produced by the evolutionary process into JavaScript arrays. Time series were recorded over the evolution process. Each row of the file contains the data as follows: time stamp, population size, global fitness, corresponding images saved flag, common error/similarity metrics between Y and $\hat{Y}$ as well as between f and $\hat{f}$ (namely mean absolute error (MAE), mean squared error (MSE), Euclidean distance, root mean squared error (RMSE), zero-normalised

cross-correlation (ZNCC), SNR, peak signal-to-noise ratio (PSNR), structural similarity (SSIM), structural dissimilarity (DSSIM)), smoothness of $\hat{Y}$ and $\hat{f}$ using total variation, and internal states of the evolutionary algorithm (e.g. probability of the various genetic operators).

The visualisation code selects user-specified columns (metrics) to make available as axes in the Parallel Coordinate Plot. Users are also offered the option to colour the lines produced according to another column (whether plotted or not). The values of that column are converted into a linear range between two user-specified colours. The tool uses the LAB colour space and HCL interpolation [25]. This results in the perceived difference in plot colour being proportional to the Euclidean distance of the colouring metric, i.e. items close to each other in the metric space will be similarly coloured in the plot. An example of this version can be seen in Fig. 6. A subsequent version added Brushing capability to the system. This implementation fades un-brushed lines to grey and leaving those of interest in their original colour. An example of this version can be seen in Fig. 7.

When (exactly) two axes are brushed, the coordinated scatterplot is also drawn. The scatterplot uses the range of the two brushed axes and only plots selected data. The Y-axis represents the lowest numbered (leftmost) axis. The colouring from the main plot is also maintained. As columns may not be in the desired order, the Parallel Coordinate Plot allows axes to be dragged left and right into the order required. The corresponding scatterplot to Fig. 7 is shown in Fig. 8. As the evolutionary process generates representative images at pre-set intervals, the points plotted are either a smaller circle, where such an image is unavailable, or a larger square where it is.

Combining these techniques, we have produced a powerful exploratory tool. It allows researchers and practitioners to gain insight into the performance of their

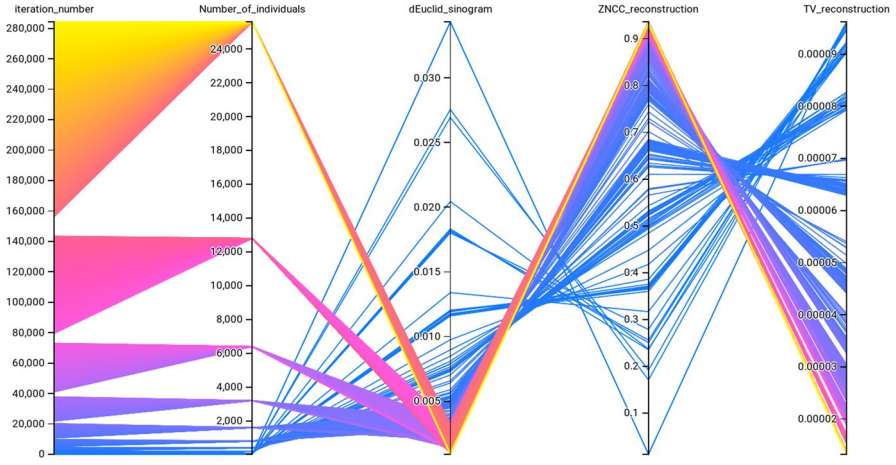

Fig. 6 Initial prototype Parallel Coordinate Plot showing an initial run of the evolutionary process. Objects are coloured according to their iteration number, included as the first axis. This is a screen-shot captured from the tool itself, the labels are clearer in the tool (Color figure online)

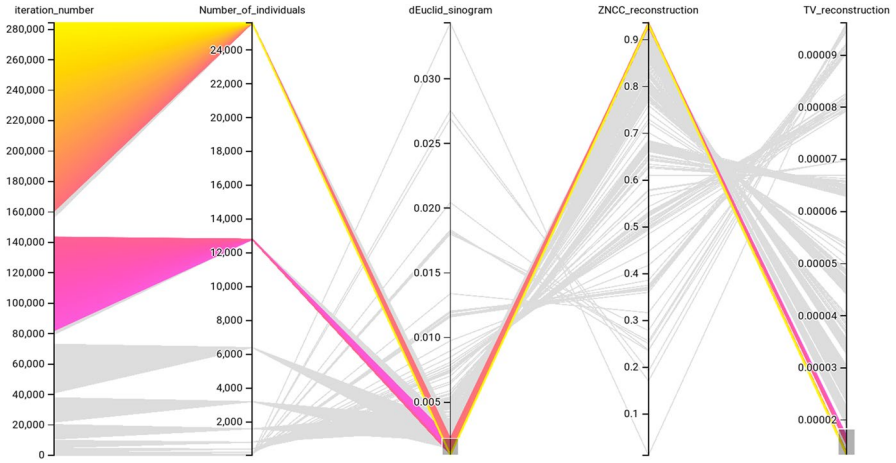

Fig. 7 The same dataset as in Fig. 6, with Brushing active on axes 3 and 5 (dEuclid_sinogram and TV_reconstruction). The ranges selected are shown by the tinted rectangle overlaid on those axes. This is a screen-shot captured from the tool itself, the labels are clearer in the tool

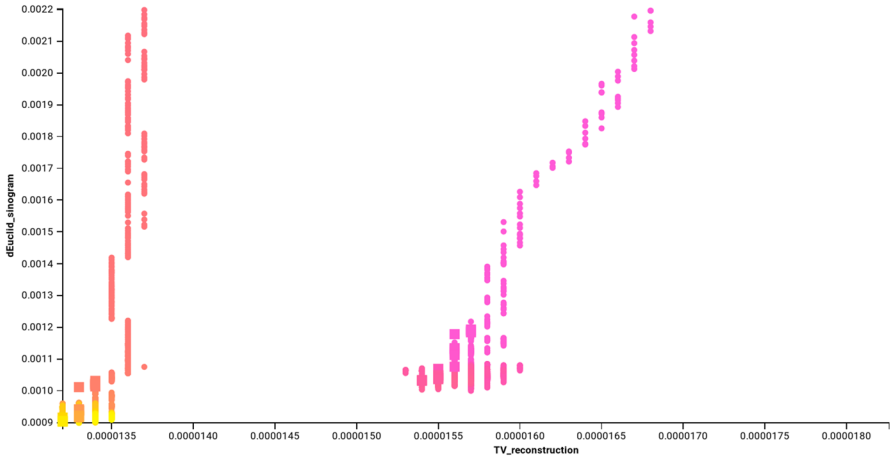

Fig. 8 The coordinated scatterplot for Fig. 7. Items are coloured as in the original figure, a smaller circle represents denotes that no image is available for that point and a larger square does. This is a screen-shot captured from the tool itself, the points and labels are clearer in the tool (Color figure online)

algorithms in an intuitive visual way. The Parallel Coordinate Plots unveil potentially masked correlations and relationships within a dataset, and the scatterplot allows reasoning about efficiency and potential tuning options. A demonstration can be seen with a modern web-browser at http://fly4pet.fpvidal.net/visualisation/.

7 Results

In past implementations [1, 5, 45–48], the final result given by the last iteration is considered as the reconstructed image. It provides a simple way to extract the answer of the optimisation problem, but it is not certain that it is the best answer that the evolutionary process provided. Our initial goal with the visualisation tool was to gain an understanding of what happens during the evolutionary process. A subsequent goal was to identify a 'good' reconstruction as quickly as possible. The ultimate goal was to develop stopping criterion dedicated to the Fly Algorithm in tomography reconstruction to automatically limit the reconstruction duration to its minimum level whilst still preserving the accuracy of the results. A good reconstruction is when the error between the simulated projection data ($\hat{Y}$) and the input data (Y) is extremely low and when the noise levels in the reconstructed volume ($\hat{f}$) are low.

Our initial assertion was that the huge amount of data generated by the evolutionary loop should not be discarded as it has the potential to actually be extremely useful to understand the reconstruction algorithm. Our initial goals were to extract the best possible solution rather than simply take the final one and to determine if any other comparable solution could have been extracted earlier on to speed-up the reconstruction time. For this purpose, we performed a reconstruction using a controlled test case and analyse the results using our visualisation. The observation data (i.e. known data) is presented in Fig. 9a. The ground-truth (i.e. unknown data) is presented in Fig. 10a. Table 1 shows the initial parameters of our Fly Algorithm for this test case.

To measure the level of similarity between two images, whether they are f and $\hat{f}$ or Y and $\hat{Y}$, we use the ZNCC:

$$ZNCC(r,t) = \frac{1}{w \times h} \sum_{y=0}^{y<h} \sum_{x=0}^{x<w} \frac{(r(x,y) - \bar{r})(t(x,y) - \bar{t})}{\sigma_r \sigma_t} \tag{6}$$

where w and h are the number of pixels along the horizontal and vertical axes in $r(x, y)$ and $t(x, y)$ respectively, $\bar{r}$ is the average pixel value of r and σ_r is standard deviation of r. The ZNCC is equal to 1 if the two images are perfectly correlated, 0

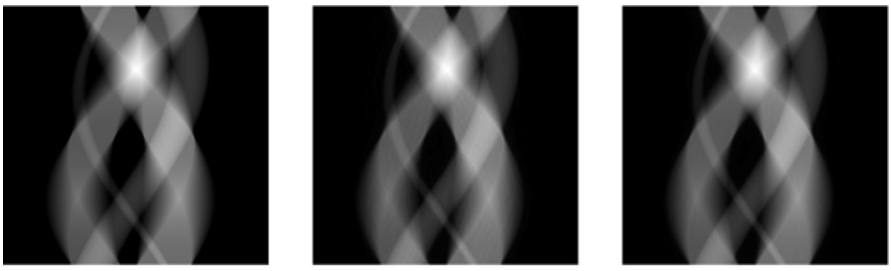

Fig. 9 Sinograms. **a** Y: input data (known), **b** $\hat{Y}_{113401}$: sinogram of the reconstruction manually selected in Fig. 15b (see green circle), **c** $\hat{Y}_{final}$: sinogram of the final reconstruction at the end of the evolution (Color figure online)

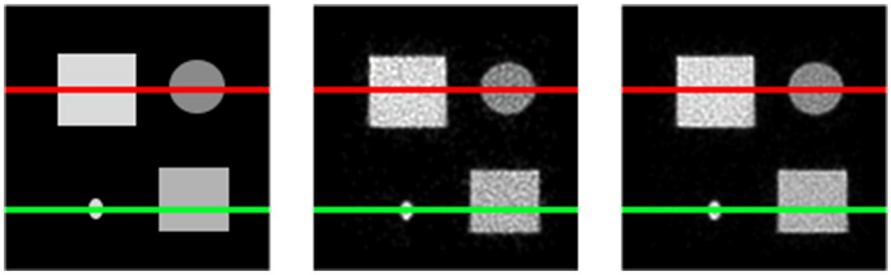

Fig. 10 Reconstructed images. **a** f: ground-truth (unknown), **b** $\hat{f}_{113401}$: reconstruction manually selected in Fig. 15b (see green circle), **c** $\hat{f}_{final}$: final reconstruction at the end of the evolution (Color figure online)

Table 1 Initial parameters of the evolutionary algorithm

Global fitness function	ℓ^2-norm
Initial population size	25
Final population size	25,600
Initial new blood probability	1/3
Initial basic mutation probability	1/3
Initial adaptive mutation probability	1/3
Selection threshold struggle	5 times in a row
Global fitness stagnation	5 times in a row (ε = 1E-2)

if they are totally uncorrelated, and -1 if they are perfectly anticorrelated (one is the negative of the other). The ZNCC is often expressed as a percentage. This image metric is very popular in image-processing and computer vision.

To measure the smoothness level of the reconstructed image $\hat{f}$, we use the total variation (TV) (also known as TV-norm):

$$\left\|\hat{f}\right\|_{TV} = \int_y \int_x \left\|\nabla_{\hat{f}}(x,y)\right\|_1 \mathrm{d}x\mathrm{d}y \tag{7}$$

where $\nabla(x, y)$ is the gradient of the corresponding image at pixel x, y. In the discrete cases, it can be computed as follows:

$$\left\|\hat{f}\right\|_{TV} = \sum_{y=0}^{y<h-1} \sum_{x=0}^{x<w-1} \sqrt{(\hat{f}(x,y) - \hat{f}(x+1,y))^2 + (\hat{f}(x,y) - \hat{f}(x,y+1))^2} \tag{8}$$

Noisy images will have a higher TV-norm than smoother images. It can be used to compute a level of quality.

Defining what is the 'best solution' is not trivial:

– Traditionally it is the final population after convergence (#284,250).

- A good candidate solution is also the one that provides the lowest global fitness ($||Y - \hat{Y}||_2^2$). In our test case, it is #269,301.
- It can also be the population that gives the lowest discrete TV seminorm of the reconstructed image ($||\hat{f}||_{TV}$). Its first occurrence is #199,101 and its last is #274,101.
- Ideally, the best solution should provide the highest ZNCC with the ground-truth (f) (ZNCC($f, \hat{f}$)), but it cannot be assessed in the reconstruction as it is not available in real cases because f is unknown. However, it can be used with test cases to analyse the behaviour of our algorithm.
- Also, a good iteration should, if possible, have a relatively small cumulative computation time up to that iteration.

We summary the performance of reconstruction at different iterations in Table 2. It presents the reconstruction cumulative computation time, the global fitness, the ZNCC between the input projections and simulated projections (ZNCC($Y, \hat{Y}$)), the TV of the reconstructed image and the ZNCC between the ground-truth and the reconstructed image. In terms of global fitness and TV, the results of the 4 iterations we selected seem to be equivalent. To assess if this is the case, we look at ZNCC($f, \hat{f}$). The values are within 0.22%. In addition, a plot combining the global fitness and ZNCC($f, \hat{f}$) is also presented (see Fig. 11). The figure shows barely any improvement, whether it is for the global fitness or ZNCC, when mitosis occurred (see pics in the graph). We can conclude that the results of the 4 iterations we selected are relatively equivalent. As we cannot distinguish between the results of the 4 iterations when looking at the global fitness and TV, we can consider the cumulative computation time (see Fig. 12). We can conclude that #199,101 is the 'best' iteration among #199,101, #269,301, #274,101, and #284,250 because its duration is the smallest: it lead to one of the best results in the smallest length of time. Spending an extra 6 minutes only marginally improved the results.

With the visualisation tool, we expect that the reconstruction time can be further reduced. The initial step is to look at how the global fitness evolves. The same dataset as Fig. 6 is plot with Brushing active on 'iteration_number' and

Table 2 Performance of reconstructions at different iterations

Iteration #	113,401	199,101	269,301	274,101	284,250
Duration (in min)	7:32	13:15	17:55	18:14	18:55
# of individuals	12,800	25,600	25,600	25,600	25,600
$\|\|Y - \hat{Y}\|\|_2^2$	10.69E-4	9.49E-4	**8.99E-4**	*9.03E-4*	***9.06E-4***
ZNCC($Y, \hat{Y}$)	***99.92%***	*99.94%*	**99.95%**	**99.95%**	*99.94%*
$\|\|\hat{f}\|\|_{TV}$	1.55E-5	**1.32E-5**	*1.33E-5*	**1.32E-5**	***1.34E-5***
ZNCC($f, \hat{f}$)	***93.20%***	**93.38%**	93.19%	*93.22%*	93.16%

The best result for each metrics is in bold, the second best in italic, and the third best in bolditalic. Iteration #113,401 has been selected by hand using our visualisation tool; #269,301 corresponds to the lowest global fitness; #199,101 corresponds to the first occurrence of the lowest TV; #274,101 corresponds to the last occurrence of the lowest TV; #284,250 corresponds to the last iteration

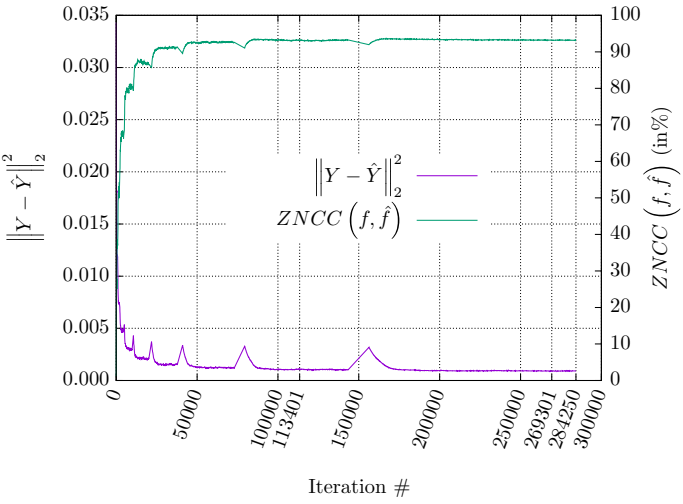

Fig. 11 Combined evolution of the global fitness and the ZNCC between the ground-truth and the reconstructed image

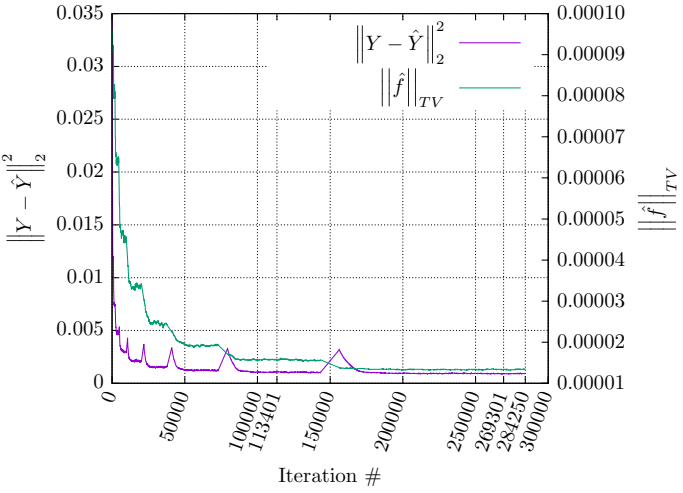

Fig. 12 Combined evolution of the global fitness and the TV of the reconstructed image

'dEuclid_sinogram' (see Fig. 13a). Note that we swapped the axes in the figure to ensure that the number of iterations corresponds to the horizontal axis, and dEuclid_sinogram to the vertical axis in the scatterplot (Fig. 13b). We observe a rapid decrease of dEuclid_sinogram with upticks when mitosis occurs. It means that the global fitness approaches its minimum at an early stage of the reconstruction process. In other words, a relatively good reconstruction is achieved quickly in terms of data fidelity between $\hat{Y}$ and Y but that other image metrics on $\hat{f}$ may

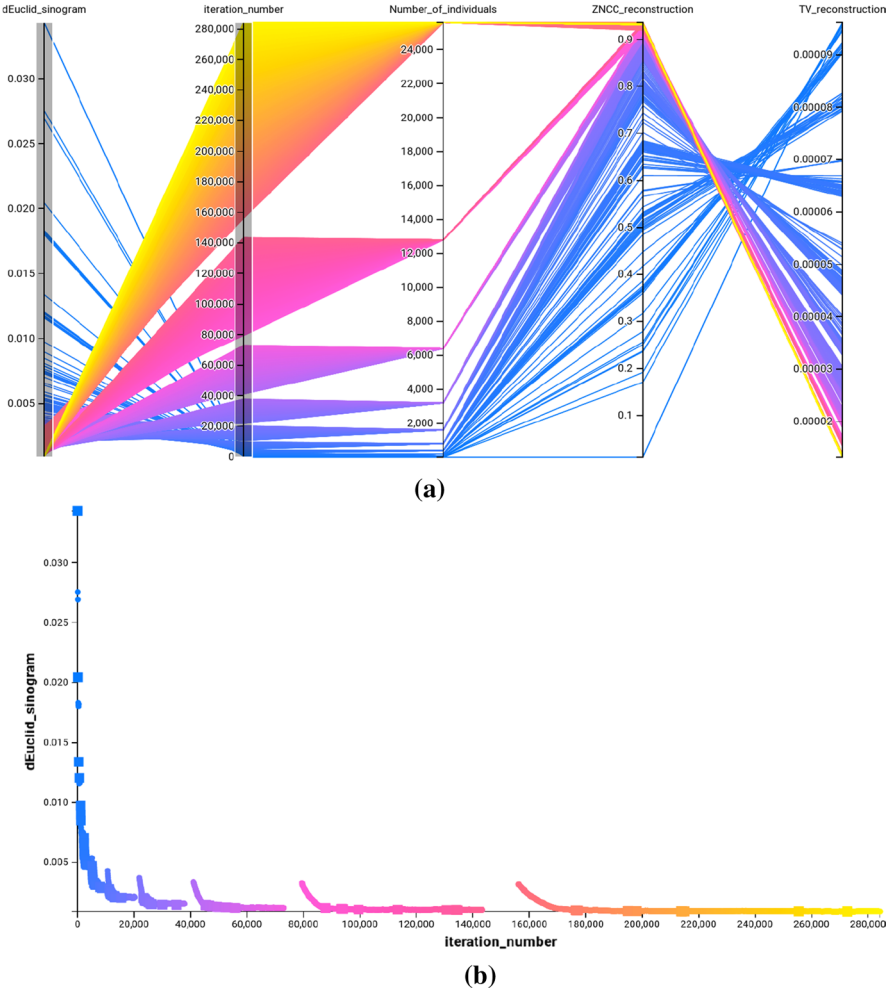

Fig. 13 Evolution of the global fitness. **a** Brushing on $||Y - \hat{Y}||_2^2$ and number of iterations, **b** corresponding scatterplot

be more relevant to decide when to stop the reconstruction or to pick a better reconstructed volume.

The same experiment is performed using TV_reconstruction rather than dEuclid_sinogram. Figure 14 shows that $||\hat{f}||_{TV}$ rapidly decreased, but a lot slower than dEuclid_sinogram. This is because the more mitosis happens, the more flies there are, resulting in less noise. However, we observe a plateau, beyond which the TV ceases to decrease significantly. This means that increasing the population size by mitosis would increase the duration without improving much the reconstruction. In this case, further investigation is needed as it indicates that the reconstruction process could have been stopped much earlier, with a lower number of flies.

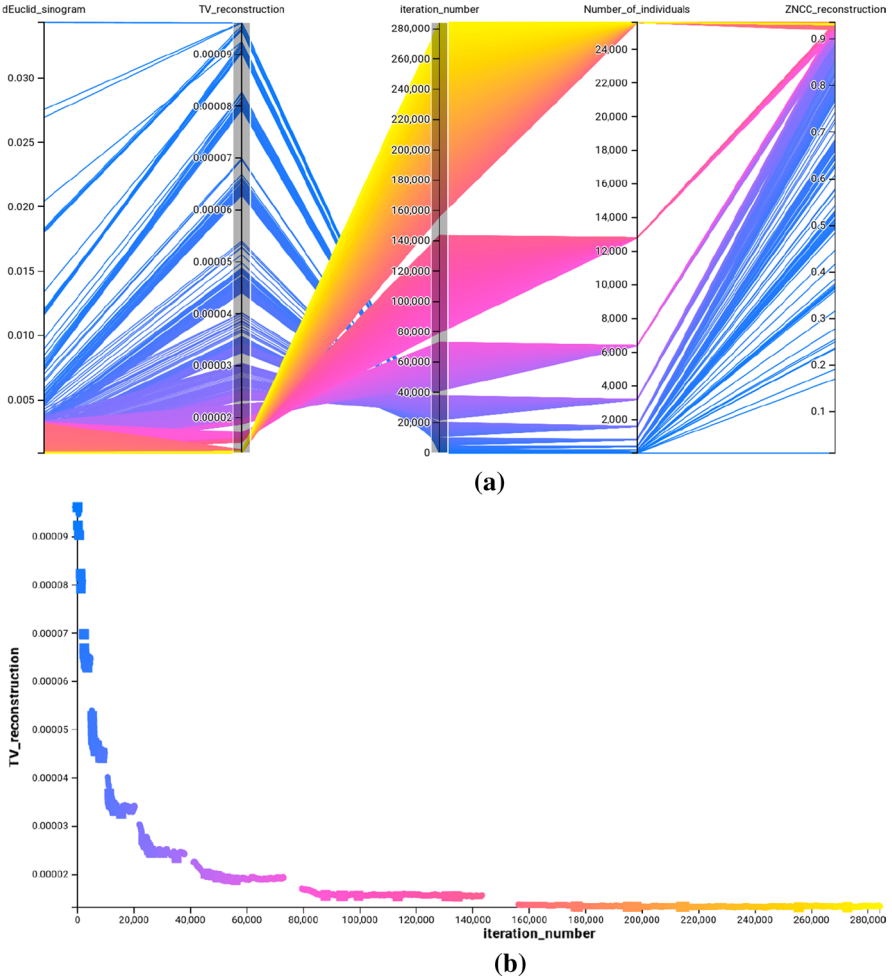

Fig. 14 Evolution of the total variation seminorm. **a** Brushing on $||\hat{f}||_{TV}$ and number of iterations, **b** corresponding scatterplot

We refine the brushed region to allow us to zoom-in on a low $||\hat{f}||_{TV}$ (see Fig. 15a). Our goal is to ascertain that $||Y - \hat{Y}||_2^2$ is still low and minimise the duration. Ideally, the best possible candidate solution will be in the lower left corner of the scatterplot (see Fig. 15b). We selected a candidate solution that is a good compromise between time and noise levels (as more iterations do not reduce $||\hat{f}||_{TV}$ much) (see green circle in Fig. 15b). It was obtained at 7:32 with 12,800 flies whereas the final candidate was obtained in 18:55 with 25,600 flies (see Table 2). It corresponds to a speedup of 2.5X.

To further validate our claim that #113,401 is a good candidate, comparable to the final one (#284,250), we now look at image data (see Figs. 9, 10 and 16). The difference, in terms of ZNCC, for $\hat{Y}$ between #113,401 and #284,250 is 0.02% (see

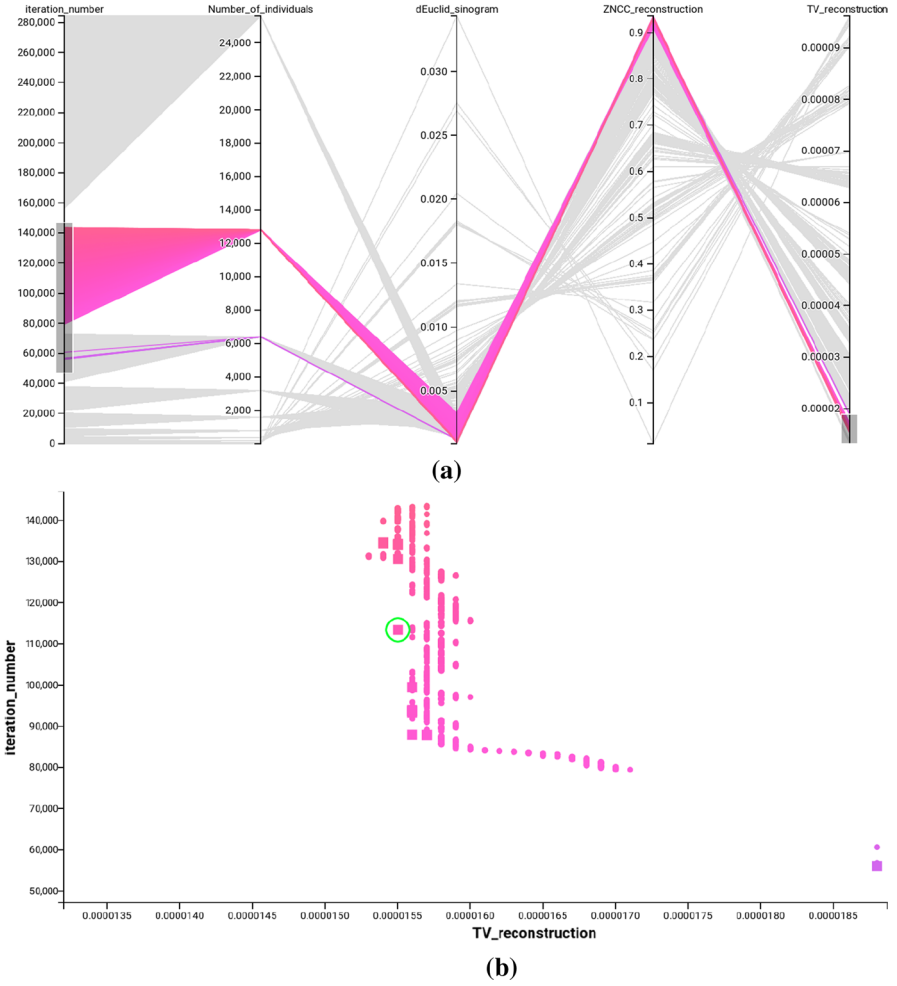

Fig. 15 Manual selection of a good candidate solution based total variation seminorm and duration. **a** Brushing on $||\hat{f}||_{TV}$ and number of iterations, **b** corresponding scatterplot. A 'good' candidate solution is circled in green (Color figure online)

Fig. 9 for the image data). This is negligible. The ZNCC of $\hat{f}$ is actually slightly smaller (by 0.04%) for #113,401 than #284,250 (see Fig. 10 for the image data). To visually assess the noise levels in #113,401 and #284,250, intensity profiles of interest are extracted. An intensity profile plots the intensity values along a line segment between to points of an image. They are shown in Fig. 16. The noise levels in #113,401 and #284,250 are very similar. We can, therefore, conclude that the extra 11:23, after iteration #113,401, did not significantly improve the reconstruction.

We further exploited these results by introducing a new stopping criterion that looks at both the global fitness and TV. The global fitness is analysed over the last 500 iterations. Using simple linear regression, the fitness values are

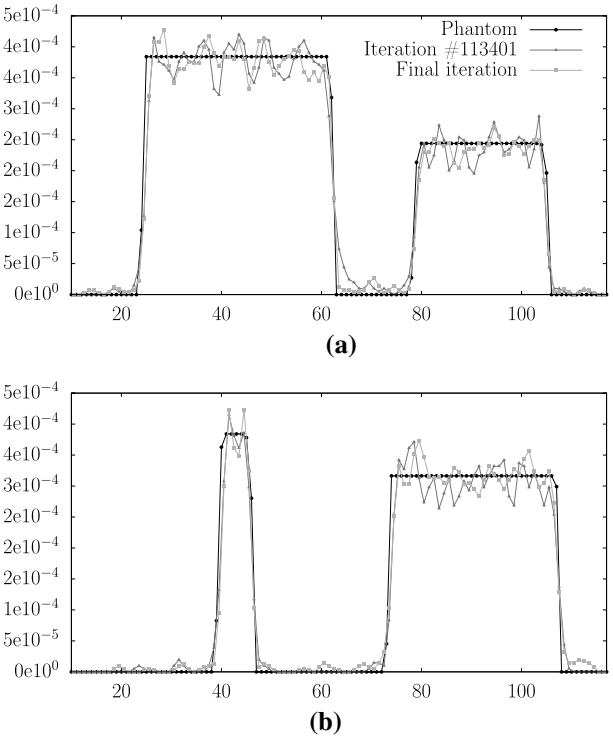

Fig. 16 Intensity profiles int the ground-truth and in the reconstructions presented in Fig. 10. **a** Intensity profiles corresponding to the red lines in Fig. 10, **b** intensity profiles corresponding to the green lines in Fig. 10 (Color figure online)

reduced to a single line, and the equation for it is extracted. When the slope is close to zero, the line is almost horizontal. It means that the global fitness has not changed much over the last 500 iterations. This process is repeated using the TV metric, again over the last 500 iterations. If the slope of both lines is below a given threshold, we deem the global fitness and TV to be stagnant. When stagnation occurs, the stopping criterion is met. To provide statistically meaningful results and due to the stochastic nature of the evolutionary algorithm, we perform 10 evolutionary reconstructions with and without our new stopping criterion. We tested this approach using three controlled test cases (see Phantoms 1, 2, and 3 in Fig. 17), therefore running 60 reconstructions in all. Figure 17 shows the reconstructed images corresponding to the median value of the total number of iterations needed for each test case.

The performance, in terms of the total number of iterations needed, global fitness, TV, and ZNCC between the reconstruction and ground-truth, is summarised in Table 3. The total number of iterations have been reduced by 68%, 67%, and 58% on average for Phantom 1, 2 and 3 respectively. It did not lead to any loss of accuracy as the ZNCC between the reconstructions and the ground-truth has

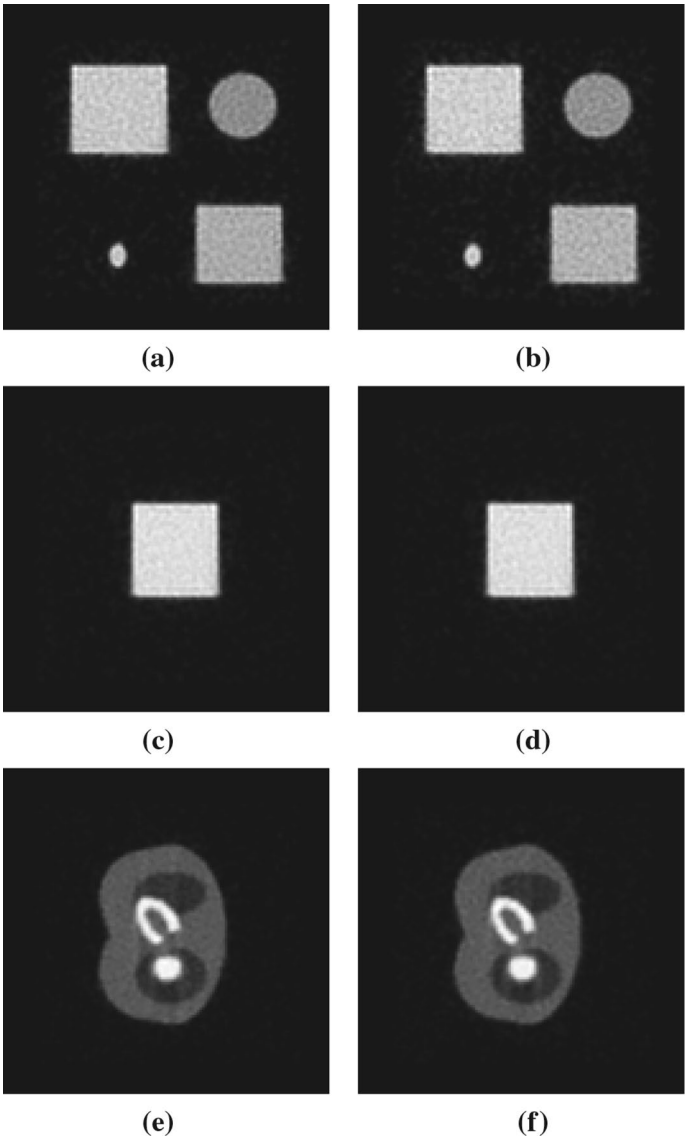

Fig. 17 Reconstruction of Phantoms 1, 2, and 3 without and with the new stopping criterion. **a** Phantom 1 without, **b** Phantom 1 with, **c** Phantom 2 without, **d** Phantom 2 with, **e** Phantom 3 without, **f** Phantom 3 with

marginally improved (by less than 0.5% for the three test cases). The TV metrics of the images reconstructed with and without the new stopping criterion are also consistent. We can conclude that the data exploration using visualisation has lead to a new stopping criterion that significantly reduces the computing time without any loss of accuracy.

Table 3 Performance comparison between the algorithm with and without the new stopping criterion using 3 test cases

		Phantom 1	Phantom 2	Phantom 3
Without	# of iterations	282590 ± 989	276320 ± 18230	282820 ± 987
	$\|\|Y - \hat{Y}\|\|_2^2$	1.09E-03 ± 5.69E-05	1.46E-03 ± 6.83E-05	8.94E-04 ± 6.88E-05
	$\|\|\hat{f}\|\|_{TV}$	1.34E-05 ± 1.43E-07	1.07E-05 ± 1.45E-07	1.60E-05 ± 1.62E-07
	ZNCC(f,$\hat{f}$)	93.23% ± 0.05%	94.34% ± 0.04%	92.59% ± 0.04%
With	# of iterations	89190 ± 5880	90370 ± 14331	117450 ± 58705
	$\|\|Y - \hat{Y}\|\|_2^2$	1.32E-03 ± 5.95E-05	1.81E-03 ± 1.30E-04	1.04E-03 ± 9.98E-05
	$\|\|\hat{f}\|\|_{TV}$	1.38E-05 ± 1.11E-07	1.13E-05 ± 2.06E-07	1.65E-05 ± 1.79E-07
	ZNCC(f,$\hat{f}$)	93.47% ± 0.05%	94.66% ± 0.07%	92.68% ± 0.06%

Each reconstruction has been performed 10 times

8 Conclusion

The research presented here relies heavily on a fully adaptive implementation of a CCEA based on the Fly Algorithm. The purpose of this algorithm is to optimise the location of 3-D points. The final set of points corresponds to the solution of the optimisation problem. We used this algorithm to solve a complex ill-posed inverse problem: tomography reconstruction in nuclear medicine. To date, the solution to the optimisation problem was extracted at the end of the evolutionary loop.

In this paper, we investigate the use of a simple but effective visualisation. It relies on Parallel Coordinate Plot, scatterplot and image display. The visualisation is used to explore the huge quantity of time series data generated by the algorithm during the optimisation loop. We focused, in particular, on metrics related to image accuracy, smoothness, and reconstruction time. We demonstrated that the final population may not be the most suitable solution and that preceding candidate solutions have to be considered to ensure that the reconstruction is accurate and not too noisy. This was not trivial as smooth images may not be accurate. This investigation allowed us to demonstrate that increasing the population size, and hence the computation time, did not necessarily lead to a significant increase in quality of the reconstruction.

This approach can be easily deployed to any evolutionary algorithm (not only Parisian Evolution) where the quality of the solution cannot be measured by a single value (usually the fitness function). It is particularly suited to multi-objective optimisation where several concurrent fitness functions are used to assess the quality of an individual. All the objectives are equally important. Multi-objective optimisation algorithms often output a set of candidate solutions (the Pareto front). Choosing which solution is the best one may not be trivial. The decision maker with expert knowledge may be able to express preferences. An interactive visualisation similar to ours has the potential to help the decision maker decide which solution(s) to pick amongst the candidates proposed by the algorithm.

We used these results to propose a new stopping criterion. It analyses the local variation in terms of global fitness and smoothness of the reconstructed image over the last 500 iterations. It allowed us to reduce the total number of iterations by almost 60% or more without any loss of accuracy.

Future work will initially include revisiting this new stopping criterion to validate it further to force an even earlier termination. We will also consider a third approach to combine visualisation and artificial evolution: interactive visualisation to steer the population in a particular area of the search space during the optimisation. The aim would be, again, to speed up the reconstruction process.

Acknowledgements The authors would like to thank HPC Wales (http://www.hpcwales.co.uk/) for providing some of the computing facilities used in this study.

Glossary

e^-	Electron
e^+	Positron
γ	Photon
β^+	Positron
f	Ground-truth (unknown radioactive concentration)
Y	Observations (known input data, e.g. sinogram)
P	System matrix/projection operator (it transforms the estimated radioactive concentration into simulated projections)
$\hat{f}$	Estimated radioactive concentration (point cloud or reconstructed image)
$\hat{Y}$	Projections simulated using the estimated radioactive concentration (simulated sinogram)
$\|\|Y - \hat{Y}\|\|_2^2$	ℓ^2-norm, also known as Euclidean distance, between Y and $\hat{Y}$
$\bar{r}$	Average pixel value of image r
σ_r	Standard deviation of pixel values of image r
$\|\|\hat{f}\|\|_{TV}$	Discrete total variation seminorm of $\hat{f}$, also known as TV-norm
$\nabla(x, y)$	Gradient of a given image at pixel x, y

References

1. Z.A. Abbood, F.P. Vidal, Basic, dual, adaptive, and directed mutation operators in the Fly algorithm, in *Biennial International Conference on Artificial Evolution (EA-2017), Paris, France* (2017), pp. 106–119
2. Z.A. Abbood, F.P. Vidal, Fly4Arts: evolutionary digital art with the Fly algorithm, in *Biennial International Conference on Artificial Evolution (EA-2017) Paris, France* (2017), p. 313

3. Z.A. Abbood, F.P. Vidal, Fly4Arts: evolutionary digital art with the Fly algorithm. ISTE Arts Sci. **17–1**(1), 11–16 (2017). https://doi.org/10.21494/ISTE.OP.2017.0177
4. Z. Ali Abbood, O. Amlal, F.P. Vidal, Evolutionary art using the fly algorithm, in *Applications of Evolutionary Computation, Lecture Notes in Computer Science*, vol. 10199 (Springer, Heidelberg, 2017), pp. 455–470. https://doi.org/10.1007/978-3-319-55849-3_30
5. Z. Ali Abbood, J. Lavauzelle, E. Lutton, J.M. Rocchisani, J. Louchet, F.P. Vidal, Voxelisation in the 3-d fly algorithm for PET. Swarm Evol. Comput. **36**, 91–105 (2017). https://doi.org/10.1016/j.swevo.2017.04.001
6. R. Amar, J. Eagan, J. Stasko, Low-level components of analytic activity in information visualization, in *IEEE Symposium on Information Visualization, 2005. INFOVIS 2005* (2005), pp. 111–117. https://doi.org/10.1109/INFVIS.2005.1532136
7. R. Amar, J. Stasko, Best paper: a knowledge task-based framework for design and evaluation of information visualizations, in *IEEE Symposium on Information Visualization* (2004), pp. 143–150. https://doi.org/10.1109/INFVIS.2004.10
8. G. Andrienko, N. Andrienko, Parallel coordinates for exploring properties of subsets, in *Second International Conference on Coordinated and Multiple Views in Exploratory Visualization, 2004. Proceedings* (IEEE, 2004), pp. 93–104
9. B. Bach, A. Spritzer, E. Lutton, J.D. Fekete, Interactive random graph generation with evolutionary algorithms, in *Graph Drawing: 20th International Symposium, GD 2012, Redmond, WA, USA*, September 19–21, 2012, Revised Selected Papers (Springer, Berlin, 2013), pp. 541–552. https://doi.org/10.1007/978-3-642-36763-2_48
10. T. Baeck, D.B. Fogel, Z. Michalewicz, (eds.), Evolutionary Computation 1: Basic Algorithms and Operators (Taylor & Francis, London, 2000). ISBN: 978-0750306645
11. O. Barrière, E. Lutton, *Experimental Analysis of a Variable Size Mono-population Cooperative-Coevolution Strategy* (Springer, Berlin, 2009), pp. 139–152. https://doi.org/10.1007/978-3-642-03211-0_12
12. O. Barrière, E. Lutton, P. Wuillemin, Bayesian network structure learning using cooperative coevolution, in *Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation, GECCO '09* (ACM, New York, 2009), pp. 755–762. https://doi.org/10.1145/1569901.1570006
13. N. Bissantz, B.A. Mair, A. Munk, A statistical stopping rule for MLEM reconstructions in PET, in *IEEE Nuclear Science Symposium Conference Record* (2008), pp. 4198–4200. https://doi.org/10.1109/NSSMIC.2008.4774207
14. C. Blum, R. Battiti, Grapheur—Business intelligence and analytics. https://www.grapheur.com/. Accessed 19 June 2018
15. P.A.N. Bosman, T. Alderliesten, Evolutionary algorithms for medical simulations: a case study in minimally-invasive vascular interventions, In *Workshops on Genetic and Evolutionary Computation*, vol. 2005 (2005), pp. 125–132. https://doi.org/10.1145/1102256.1102286
16. M. Bostock, D3.js—Data Driven Documents—v. 5.5.0. https://www.d3js.org/. Accessed 19 June 2018
17. A.M. Boumaza, J. Louchet, *Mobile Robot Sensor Fusion Using Flies* (Springer, Berlin, 2003), pp. 357–367. https://doi.org/10.1007/3-540-36605-9_33
18. A. Bousquet, J. Louchet, J.M. Rocchisani, Fully three-dimensional tomographic evolutionary reconstruction in nuclear medicine, in *Artificial Evolution: 8th International Conference, Evolution Artificielle, EA 2007, Tours, France*, October 29–31, 2007, Revised Selected Papers, Lecture Notes in Computer Science, vol. 4926 (Springer, Berlin 2008), pp. 231–242. https://doi.org/10.1007/978-3-540-79305-2_20
19. M. Brunato, R. Battiti, Grapheur: a software architecture for reactive and interactive optimization, in *Learning and Intelligent Optimization*, vol. 6073, Lecture Notes in Computer Science, ed. by C. Blum, R. Battiti (Springer, Berlin, 2010), pp. 232–246
20. S. Cagnoni, A.B. Dobrzeniecki, R. Poli, J.C. Yanch, Genetic algorithm-based interactive segmentation of 3D medical images. Image Vis. Comput. **17**(12), 881–895 (1999). https://doi.org/10.1016/S0262-8856(98)00166-8
21. K. Clark, B. Vendt, K. Smith, J. Freymann, J. Kirby, P. Koppel, S. Moore, S. Phillips, D. Maffitt, M. Pringle, L. Tarbox, F. Prior, The cancer imaging archive (TCIA): maintaining and operating a public information repository. J. Digit. Imaging **26**(6), 1045–1057 (2013). https://doi.org/10.1007/s10278-013-9622-7

22. P. Collet, J. Louchet, Applications in the Processing of Signals and Images, Chapter 2, in *Artificial Evolution and the Parisian Approach* (Wiley, 2010), pp. 15–44. https://doi.org/10.1002/9780470611319.ch2
23. F.H. Fahey, Data acquisition in PET imaging. J. Nucl. Med. Technol. **30**(2), 39–49 (2002)
24. Z. Halim, T. Muhammad, Quantifying and optimizing visualization: an evolutionary computing-based approach. Inf. Sci. **385–386**(Supplement C), 284–313 (2017). https://doi.org/10.1016/j.ins.2016.12.035
25. A. Hanbury, Constructing cylindrical coordinate colour spaces. Pattern Recognit. Lett. **29**(4), 494–500 (2008)
26. J. Heer, B. Shneiderman, Interactive dynamics for visual analysis. Queue **10**(2), 30 (2012)
27. H.M. Hudson, R.S. Larkin, Accelerated image reconstruction using ordered subsets of projection data. IEEE Trans. Med. Imaging **13**(4), 601–609 (1994). https://doi.org/10.1109/42.363108
28. A. Inselberg, The plane with parallel coordinates. Vis. Comput. **1**(2), 69–91 (1985). https://doi.org/10.1007/BF01898350
29. B. Kaufmann, J. Louchet, E. Lutton, Hand posture recognition using real-time artificial evolution, in *Evolutionary Computation in Image Analysis and Signal Processing, EvoApplications 2010, Part I, LNCS 6024,C*, ed. by D. Chio et al. (Springer, 2010), pp. 251–260. 7th–9th April, Istanbul Technical University, Istanbul, Turkey
30. A. Kerren, T. Egger, Eavis: a visualization tool for evolutionary algorithms, in *2005 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC'05)* (2005), pp. 299–301. https://doi.org/10.1109/VLHCC.2005.33
31. Y. Landrin-Schweitzer, P. Collet, E. Lutton, Introducing lateral thinking in search engines. Genet. Program. Evol. Mach. **7**(1), 9–31 (2006). https://doi.org/10.1007/s10710-006-7008-z
32. B. Lee, P. Isenberg, N.H. Riche, S. Carpendale, Beyond mouse and keyboard: expanding design considerations for information visualization interactions. IEEE Trans. Vis. Comput. Graph. **18**(12), 2689–2698 (2012)
33. J. Louchet, Stereo analysis using individual evolution strategy, in *15th International Conference on Pattern Recognition, 2000. Proceedings*, vol. 1 (2000), pp. 908–911. https://doi.org/10.1109/ICPR.2000.905580
34. J. Louchet, Using an individual evolution strategy for stereovision. Genet. Program. Evol. Mach. **2**(2), 101–109 (2001). https://doi.org/10.1023/A:1011544128842
35. J. Louchet, M. Guyon, M.J. Lesot, A. Boumaza, Dynamic flies: a new pattern recognition tool applied to stereo sequence processing. Pattern Recognit. Lett. **23**(1), 335–345 (2002). https://doi.org/10.1016/S0167-8655(01)00129-5
36. E. Lutton, N. Perrot, A. Tonda, Model analysis and visualization, in *Evolutionary Algorithms for Food Science and Technology* (Wiley, 2016), pp. 33–55. https://doi.org/10.1002/9781119136828.ch3
37. A.R. Martin, M.O. Ward, High dimensional brushing for interactive exploration of multivariate data, in *Proceedings of the 6th Conference on Visualization'95* (IEEE Computer Society, 1995), p. 271
38. H. Pohlheim, Visualization of evolutionary algorithms-set of standard techniques and multidimensional visualization, in *Proceedings of the 1st Annual Conference on Genetic and Evolutionary Computation-Volume 1, GECCO'99* (Morgan Kaufmann Publishers Inc., San Francisco, 1999), pp. 533–540
39. J. Qi, R.M. Leahy, Iterative reconstruction techniques in emission computed tomography. Phys. Med. Biol. **51**(15), R541 (2006). https://doi.org/10.1088/0031-9155/51/15/R01
40. S. Ribecca, The Data Visualisation Catalogue. https://www.datavizcatalogue.com. Accessed 19 June 2018
41. J.C. Roberts, State of the art: coordinated and multiple views in exploratory visualization, in *Fifth International Conference on Coordinated and Multiple Views in Exploratory Visualization, 2007. CMV'07* (IEEE, 2007), pp. 61–71
42. E. Sapin, J. Louchet, E. Lutton, The fly algorithm revisited-adaptation to CMOS image sensors, in *IJCCI* (2009), pp. 224–229
43. L. Shepp, Y. Vardi, Maximum likelihood reconstruction for emission tomography. IEEE Trans. Med. Imaging **1**(2), 113–122 (1982). https://doi.org/10.1109/TMI.1982.4307558
44. Tableau Software: Business Intelligence and Analytics. https://www.Tableau.com/. Accessed 19 June 2018

45. F.P. Vidal, D. Lazaro-Ponthus, S. Legoupil, J. Louchet, É. Lutton, J.M. Rocchisani, Artificial evolution for 3D PET reconstruction, in *Proceedings of the 9th International Conference on Artificial Evolution (EA'09), Lecture Notes in Computer Science*, vol. 5975 (Springer, Heidelberg, 2009), pp. 37–48. https://doi.org/10.1007/978-3-642-14156-0_4
46. F.P. Vidal, J. Louchet, E. Lutton, J. Rocchisani, PET reconstruction using a cooperative coevolution strategy in LOR space, in *IEEE Nuclear Science Symposium Conference Record* (IEEE, 2009), pp. 3363–3366. https://doi.org/10.1109/NSSMIC.2009.5401758
47. F.P. Vidal, J. Louchet, J. Rocchisani, E. Lutton, New genetic operators in the Fly algorithm: application to medical PET image reconstruction, in *Applications of Evolutionary Computation, Lecture Notes in Computer Science*, vol. 6024 (Springer, Heidelberg, 2010), pp. 292–301. https://doi.org/10.1007/978-3-642-12239-2_30
48. F.P. Vidal, E. Lutton, J. Louchet, J. Rocchisani, Threshold selection, mitosis and dual mutation in cooperative coevolution: application to medical 3D tomography, in *International Conference on Parallel Problem Solving From Nature (PPSN'10), Lecture Notes in Computer Science*, vol. 6238 (Springer, Heidelberg, 2010), pp. 414–423. https://doi.org/10.1007/978-3-642-15844-5_42
49. F.P. Vidal, P. Villard, E. Lutton, Tuning of patient specific deformable models using an adaptive evolutionary optimization strategy. IEEE Trans. Biomed. Eng. **59**(10), 2942–2949 (2012). https://doi.org/10.1109/TBME.2012.2213251
50. K. Völk, J.F. Miller, S.L. Smith, Multiple network CGP for the classification of mammograms, in *EvoWorkshops 2009, LNCS*, vol. 5484 (Springer, 2009), pp. 405–413. https://doi.org/10.1007/978-3-642-01129-0_45
51. S. Wehrend, C. Lewis, A problem-oriented classification of visualization techniques, in *Proceedings of the First IEEE Conference on Visualization, 1990. Visualization'90* (IEEE, 1990), pp. 139–143
52. A.S. Wu, K.A.D. Jong, D.S. Burke, J.J. Grefenstette, C.L. Ramsey, Visual analysis of evolutionary algorithms, in *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406)*, vol. 2 (1999), p. 1425. https://doi.org/10.1109/CEC.1999.782649
53. H.C. Wu, C.T. Sun, S.S. Lee, Visualization of evolutionary computation processes from a population perspective. Intell. Data Anal. **8**(6), 543–561 (2004)
54. Z. Xie, S. Huang, M.O. Ward, E.A. Rundensteiner, Exploratory visualization of multivariate data with variable quality, in *2006 IEEE Symposium on Visual Analytics Science And Technology* (IEEE, 2006), pp. 183–190
55. H. Zhou, X. Yuan, H. Qu, W. Cui, B. Chen, Visual clustering in parallel coordinates, in *Computer Graphics Forum*, vol. 27 (Wiley Online Library, 2008), pp. 1047–1054

Affiliations

Cameron C. Gray[1] · Shatha F. Al-Maliki[1,2] · Franck P. Vidal[1]

Cameron C. Gray
c.gray@bangor.ac.uk

Shatha F. Al-Maliki
shatha.f.maliki@bangor.ac.uk

[1] School of Computer Science, Bangor University, Dean Street, Bangor, Gwynedd LL57 1UT, UK

[2] Department of Computer Science, Basrh University, Basra, Iraq