



HAL
open science

Visual analysis of population scatterplots

Evelyne Lutton, Julie Fouquier, Nathalie Perrot, Jean Louchet, Jean-Daniel Fekete

► **To cite this version:**

Evelyne Lutton, Julie Fouquier, Nathalie Perrot, Jean Louchet, Jean-Daniel Fekete. Visual analysis of population scatterplots. 10th Biannual International Conference on Artificial Evolution (EA-2011), Oct 2011, Angers, France. hal-00642307

HAL Id: hal-00642307

<https://inria.hal.science/hal-00642307>

Submitted on 17 Nov 2011

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Visual Analysis of population scatterplots

Evelyne Lutton¹, Julie Foucquier², Nathalie Perrot²,
Jean Louchet³, and Jean-Daniel Fekete¹

¹ AVIZ Team, INRIA Saclay - Ile-de-France,
Bat 490, Université Paris-Sud, 91405 ORSAY Cedex, France.

`Evelyne.Lutton@inria.fr`, `Jean-Daniel.Fekete@inria.fr`

² UMR782 Génie et Microbiologie des Procédés Alimentaires.
AgroParisTech, INRA, 78850 Thiverval-Grignon, France.

`Julie.Foucquier@grignon.inra.fr`, `nathalie.perrot@grignon.inra.fr`

³ Artenia, 24 rue Gay-Lussac, 92320 Châtillon, France
`Jean.Louchet@gmail.com`

Abstract. We investigate how visual analytic tools can deal with the huge amount of data produced during the run of an evolutionary algorithm. We show, on toy examples and on two real life problems, how a multidimensional data visualisation tool like ScatterDice/GraphDice can be easily used for analysing raw output data produced along the run of an evolutionary algorithm. Visual interpretation of population data is not used very often by the EA community for experimental analysis. We show here that this approach may yield additional high level information that is hardly accessible through conventional computation.

Keywords : Optimisation, Artificial Evolution, Genetic algorithms, Visual Analytics, Experimental analysis of algorithms, parameter tuning, fitness landscape visualisation.

1 Introduction

Experimental analysis is an important aspect of research in Evolutionary Computation, both in the development of new algorithms and in the understanding of the internal mechanisms of artificial evolution. It is particularly difficult to deal with the huge amount of information that can be collected during the run of an evolutionary algorithm, as the information is usually highly multidimensional, and combines continuous, discrete and symbolic data.

Visualisation utilities associated with available EA software usually display fitness curves, convergence diagrams, and various statistics, but they more rarely allow to visualise raw genomic data. The importance of efficient visualisation tools for EAs is not a new issue [18], and various solutions have been proposed. The methods are commonly split into two types of visualisation tools: on-line tools, that display a set of monitoring curves during an EA run (e.g. fitness of the best individual, statistics, diversity), and off-line tools, that do a “post-mortem” analysis with more sophisticated results. However, important, difficult issues remain unsolved:

II

- How to visualise an individual. This task is particularly complex when the search space is large and multidimensional, or when the genome combines symbolic and numerical (discrete and/or continuous) values. The particular case of genetic programming has also been considered[7]. Existing solutions are usually problem dependent, and may call for the display of phenotypes (signals, images, sounds, graphs, networks, etc ...).
- How to visualise a population, i.e. a possibly large set of multidimensional points. The ability to efficiently visualise fitness landscapes is still a challenge: approaches like fitness-distance correlation only give a partial answer.
- How to visualise the history and evolutionary mechanisms. This issue is important, as visualising various statistics about an evolving population may not be enough to understand some of the complex or hidden mechanisms of EAs, like the action of operators. Being able to follow the transmission of genetic material inside a population has been partially addressed by schemata analysis, however there is still a strong need when dealing with continuous landscapes.
- How to visualise the result in non-standard EAs. A good example of this is the case of multi-objective EAs, as the growing size of the problem EA are able to solve lead to outputs made of large, high dimensional Pareto datasets. There is a strong need of efficient visualisation tools, that may help to monitor (on line or even interactively) multidimensional Pareto fronts [13].

In this paper, we investigate recent tools developed by the visual analytics community, that may provide efficient and generic answers to some of the previous challenges. The paper is organised as follows. A short review of existing visualisation systems for EAs is given in Section 2. Section 3 presents the ScatterDice / GraphDice tool. An analysis of population clouds using ScatterDice/GraphDice is developed in Section 4 for some classical test functions. Tests are based on the EASEA language[4]⁴. Two real life examples are then presented in Section 5, and as a conclusion Section 6 sketches future developments for a GraphDice version adapted to EA visualisation.

2 Visualisation of EA data

2.1 On-line visualisation

Almost any evolutionary software proposes nowadays its own on-line display facilities. It is often reduced to visualise how the best fitness value evolves along generations. We give below some examples that provide additional features.

Bedau and Bullock[1, 3] show the importance of tracking evolutionary activity via plots that visualise the history at different levels. For instance, genotype's activity corresponds to the frequency of a given genotype in a population, which appears, increases or decreases along generations, forming what they call "waves".

Pohlheim[15] proposed in 1999 a visualisation system adapted to the Genetic and Evolutionary Algorithm Toolbox for Matlab - GEATbx[14]. His system allows various

⁴ The software is available at <http://sourceforge.net/projects/easea/>

visualisation modes, and gives for instance the current state of a population (one generation), visualises a run (all the generations), or different runs for comparisons. Additionally he copes with the problem of visualising high-dimensional data using multidimensional scaling (reduction to a 2D visualisation that preserves distance relationships), and uses a 2D representation to show the paths followed by the best individual of every generation.

Kerren and Egger in 2005 [6] developed a Java-based on-line visualisation tool, EAVis, in which it is possible to embed an application-dependent Phenotype View.

Collins in 2003 [5] provided a survey chapter on this topic and identified some directions for future research. His main conclusion concerns the strong need for flexible visualisation environments, as he considered that current solutions were still too problem dependent.

2.2 Off-line visualisation

Off-line visualisation systems allow displaying more data, including multidimensional data, which is one of the important issues in current visualisation systems. Spears [19] provided in 1999 a brief overview of multidimensional visualisation techniques for visualising EAs, such as the use of colour, glyphs, parallel coordinates or coordinates projections. For discrete data, Routen in 1994 [16] suggested to adopt a modified Hinton diagram⁵ in which a rectangle represents a chromosome and its size is determined by the fitness of the chromosome. Let us give below a list of some off-line systems:

- William Shine and Christoph Eick[18] describe the features of a GA-visualization environment that uses quadcodes to generate search space coverage maps, employs 2D-distance maps to visualize convergence, and uses contour maps to visualize fitness.
- The VIS system [21] proposed in 1999 allows a navigation at various levels of detail, and manages transitions between related data levels. The visualisation of the most detailed level is based on ad-hoc representations (bar codes, colors, alleles frequencies) but does not allow visualising multidimensional continuous genomes.
- Emma Hart [10] proposed GAVEL in 2001, an off-line visualisation system adapted to generational GAs, that provides a means to understand how crossover and mutation operations assemble the optimal solution, and a way to trace the history of user-selected sets of alleles. It allows a display of the complete history across all generations of chromosomes, individual genes, and schemata.
- Marian Mach [12] presented in 2002 a simple and interactive “post-mortem” GA visualising tool focused on the visualisation of multidimensional data via 1D projections.

Annie Wu [21], who developed the VIS system, gave the following list of desirable tasks for visualisation systems: (*a*) to examine individuals and their encodings in detail, (*b*) to trace the source and survival of building blocks or partial solutions, (*c*) to trace

⁵ A Hinton diagram provides a qualitative display of the values in a matrix. Each value is represented by a square whose size is related to the magnitude, and color indicates sign.

family trees to examine the effects of genetic operators, *(d)* to examine populations for convergence, speciation, etc, *(e)* to trace gross population statistics and trends to move freely in time and through populations.

The genome representation issue, item *(a)*, is perhaps the most complex one, and as we have seen above, various solutions have been proposed, depending if we are dealing with continuous or discrete genomes. For Genetic Programming, the question is even more complex: a solution proposed by Jason Daida [7] in 2005 consists in visualizing big tree structures as large sized graphs.

The issue addressed by the GAVEL system, that appears as very challenging in off-line systems and that is mentioned by Annie Wu as items *(c)* and *(d)*, is to be able to trace the history of individuals. Spears [19] mentioned also that being able to track the origin of the fittest individual per generation is a very important issue for parameter tuning.

The question of family trees visualisation has more recently been considered by Zbigniew Walczak in 2005 in a short chapter [20] where he proposed to visualise evolutionary processes using graph drawing software.

3 ScatterDice / GraphDice

Visual analytics is a multidisciplinary field that integrates various sophisticated computational tools with innovative interactive techniques and visual representations to facilitate human interpretation of raw data. We present in this section a tool developed by researchers in this field, that seems to answer in a generic way to some needs identified in the previous sections.

ScatterDice[8] is a multidimensional visual exploration tool, that enables the user to navigate in a multidimensional set via simple 2D projections, organised as scatterplot matrices. The visual coherence between various projections is based on animated 3D transitions. A scatterplot matrix presents an overview of the possible configurations, thumbnails of the scatterplots, and support for interactive navigation in the multidimensional space. Various queries can be built using bounding volumes in the dataset, sculpting the query from different viewpoints to become more and more refined. Furthermore, the dimensions in the navigation space can be reordered, manually or automatically, to highlight salient correlations and differences among them⁶.

A recent evolution of ScatterDice using the same principles but with many additional features is GraphDice[2]. It allows reading the same type of data (.csv files), and other more sophisticated formats, as it also embeds graph visualisation utilities⁷.

4 Analysing successive populations

ScatterDice or GraphDice can be used to visualise data collected during the run of an EA[11]. At each generation, the content of the current population can be written

⁶ A demo of ScatterDice can be launched from <http://www.aviz.fr/~fekete/scatterdice/>, it accepts standard .csv files (although it may be necessary to add a second line after the header giving the data type for each column - INT, STR, REAL, etc).

⁷ A demo of GraphDice is also accessible at <http://www.aviz.fr/graphdice/>

into a “.csv” file as on figure 1, creating what can be called a “cloud” of successive populations. This cloud of multidimensional points is visualised using ScatterDice or GraphDice, to produce various, sometimes unusual, viewpoints.

```

Generation;Fitness;x[0];sigma[0];x[1];sigma[1]
INT;DOUBLE;DOUBLE;DOUBLE;DOUBLE;DOUBLE
0;3.75447;-0.12508;0.195626;0.524069;0.255402
0;1.17484;-0.573358;0.142053;0.924887;0.392851
0;2.28066;-0.533583;0.183641;0.546523;0.461643
0;1.92319;-0.70052;0.338246;0.582533;0.406443
0;2.75539;0.784538;0.182157;-0.940648;0.383136
0;3.08894;-0.770051;0.190012;-0.840338;0.359992
0;2.30766;0.380979;0.124581;0.0379446;0.469388
0;3.30957;-0.704403;0.453222;0.208484;0.182612
...

```

Fig. 1. A simple .csv file collected during a run (minimisation of the 2D Weierstrass function $H = 0.2$)

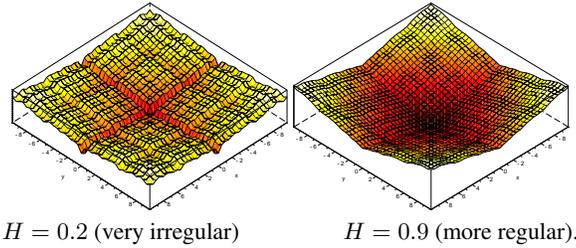


Fig. 2. 2D Weierstrass functions with Hölder Exponent H .

The test functions used in the experiments below are the following:

- Weierstrass functions (see figure 2) defined in a space of dimension 2, of Hölder exponents $H = 0.2$ (very irregular) and $H = 0.9$ (more regular).

$$f(x, y) = \sum_{n=-\infty}^{+\infty} 2^{-nH} (1 - \cos 2^n x) + \sum_{n=-\infty}^{+\infty} 2^{-nH} (1 - \cos 2^n y)$$

- Rosenbrock function⁸ in a space of dimension 10.

$$f(x_1, \dots, x_{10}) = \sum_{n=1}^9 100(x_n^2 - x_{n+1})^2 + (1 - x_n)^2$$

The genetic engine is a simple generational algorithm on R^n using tournament selection, geometric (barycentric) crossover and log-normal self-adaptive mutation. Additional dimensions are thus considered in the search space, the σ_i values, that represent the mutation radius for each coordinate x_i . The population size is 100 and the algorithm runs for 100 generations. This genetic engine is available in the sample programs (weierstrass.ez) distributed with the EASEA software.

⁸ See for instance http://en.wikipedia.org/wiki/Rosenbrock_function

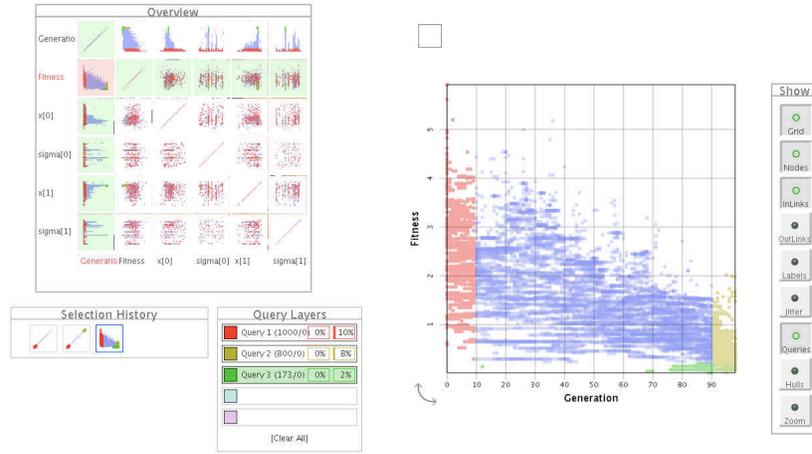


Fig. 3. 2D Weierstrass function of Hölder exponent 0.2. Scatterplot and Fitness versus generation view. Red points correspond to the first 10 generations, yellow points, to the 10 last ones, and green points, to the best fitness areas.

A visualisation of the population cloud for the 2D Weierstrass function with dimension 0.2 is given on figure 3. The scatterplot matrix on the left of the figure, gives an overview of the possible visualisations. The columns and lines of this matrix can be dragged and dropped as wished by the user. A default order is proposed, based on an algorithm that reduces “clutter”[2]. On the right, a detailed view is given, corresponding to the selected cell of the matrix (highlighted in red), on which some queries have been visualised: in red, the points corresponding to the first 10 generations of the run, in yellow, the last 10 generations, and in green, the best fitness points. The queries are organised as layers, the “Query layers” window gives the details of the three queries, with some additional measurements (percentage of selected points, and percentage of selected edges if a graph is visualised : GraphDice actually considers a set of points as a degenerate graph, made of a collection of nodes with no edges). Bottom left, a “Selection History” window shows how the queries have been sculpted: queries 1 and 2 have been activated on the Generation versus Generation view, i.e. the top left plot of the scatterplot matrix, while query 3 has been made on the fitness versus generation plot. On the extreme right of the window, a toolbar proposes various visualisation options (the “Show” window), for instance “Grid” activates a grid on the dataset, “Labels” allow to display attributes attached to a point. It is thus possible for instance to identify different runs of an EA using a label, and use this option to separate the data when needed. “Hull” displays a convex hull for each query, and “Zoom” activates an automatic zoom focused on the selected data.

The Fitness versus Generation plot, that displays the whole set of individuals generated along the evolution, provides additional information about the distribution of successive populations. When observed from a different viewpoint, for instance according to x_0 and x_1 or to σ_0 and σ_1 like in figure 4, it can be noticed that the population diver-

sity decreases slowly, and converge toward a point of the 2D plane, while stabilising on some mutation parameters. The first generations (red points of the query 1) are spread in a rather uniform way on the whole search space, while the last ones (yellow points) are concentrated in the areas of best fitness (green points). A green point appears rather early (see generation 12 on the Fitness versus Generation plot), but green points start to multiply rather late (from generation 68) in the evolution.

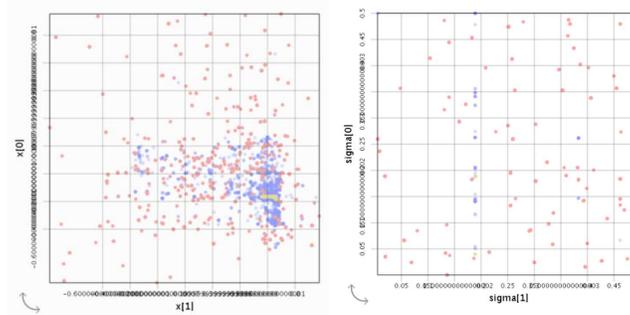


Fig. 4. 2D Weierstrass function $H = 0.2$. Left, projection on the 2D plane (x_0, x_1) . Right, parameters σ_0 and σ_1 . Points in red belong to generations 0 to 10, points in yellow to 90 to 100.

Figure 5 gives the same view as figure 3 but for a 2D Weierstrass function of dimension 0.9. This function is much more regular than the previous one. Visually, it seems obvious that the population is able to converge more rapidly.

Figure 6 gives an overview of the visualisation window for a dimension 10 space. Once again, as the function is more regular, the population seems to converge rapidly, even if it uses the same parameters as for the Weierstrass functions, in a search space with higher dimension.

5 Analysing an evolved population

GraphDice can also be used a more conventional way, in order to visualise the final population of an evolutionary algorithm. In the sequel, we present the analysis that can be made on two real life problems. The first one is a classical steady-state genetic algorithm, for which visual inspection of the evolved population allows identifying a linear dependency between the variables. The second one is an example of visualisation for a cooperative-coevolution algorithm, the fly algorithm, for which a major part of the evolved population represents the searched solution.

VIII

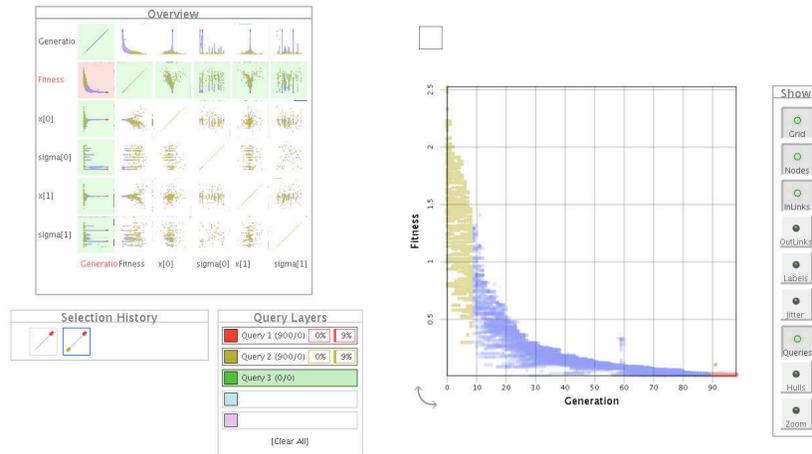


Fig. 5. 2D Weierstrass function of Hölder exponent 0.9. Yellow points correspond to the first generations, red points to the last generations.

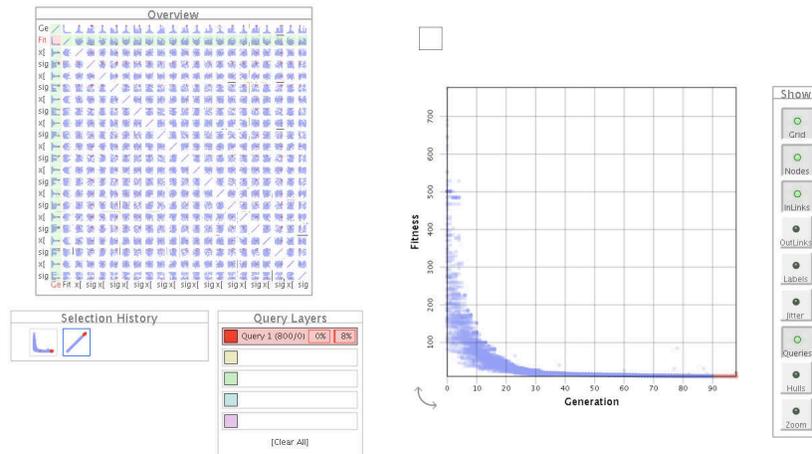


Fig. 6. Rosenbrock function in dimension 10. Red points correspond to the 10 last generations.

5.1 Classical optimisation in R^2

The optimisation problem considered here is related to modelling the behaviour of an emulsion of milk, proteins and fat. A model of whey protein⁹ and casein micelles adsorption on fat droplets has been built, based on geometrical reasoning[9]. The resulting differential equations depend on two unknown parameters k_{cas} and k_{wp} . An evolutionary algorithm has been used to learn these parameters using experimental data. The final population has been visualised using GraphDice.

A linear dependency between k_{cas} and k_{wp} has been made obvious on Figure 7: best fitted points of the final population are centred on the line (red points); when the fitness decreases, the dependency becomes less strict (yellow and green points) and the corresponding points are distributed around the diagonal line. It seems thus more convenient to deal with a one dimensional problem, i.e. to find the best ratio k_{cas}/k_{wp} . This experimental evidence sheds a new light on the *a priori* geometrical model, which needs now to be refined.

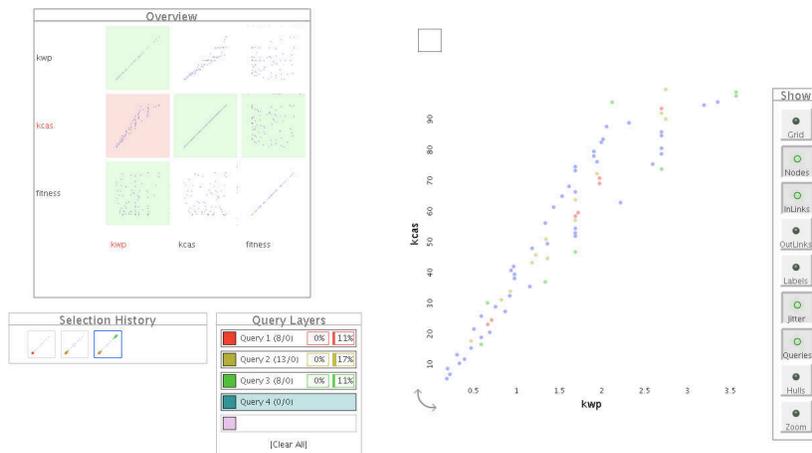


Fig. 7. Red points correspond to the 11% best fitness individuals, yellow points, to the 17% “next” best points, and green points, to the 11% worst fitness points. Queries have been interactively sculpted on the fitness versus fitness view. On the k_{cas} versus k_{wp} view, a linear dependence between the two parameters has been made obvious.

5.2 Cooperative Coevolution of a set of 3D points

In the computer vision domain, Cooperative Coevolution algorithms (CCEAs) have been applied to stereovision, to produce the fly algorithm[17]. This algorithm evolves a

⁹ Whey is left over when milk coagulates and contains everything that is soluble from milk.

X

population of 3-D points, the flies, so that the population matches the shapes of the objects on the scene. It is a cooperative coevolution in the sense that the searched solution is represented by the whole population rather than by the single best individual.

An individual of the population, a “fly”, is defined as a 3-D point with coordinates (x,y,z) . If the fly is on the surface of an opaque object, then the corresponding pixels in the two images will normally have highly similar neighbourhoods. Conversely, if the fly is not on the surface of an object, their close neighbourhoods will usually be poorly correlated. The fitness function exploits this property and evaluates the degree of similarity of the pixel neighbourhoods of the projections of the fly, giving higher fitness values to those probably lying on objects surfaces.

GraphDice allows visualising a population of flies, and rapidly provides various viewpoints on the evolved data set (figures 9 and 10), in addition to *ad hoc* visualisations (figure 8)



Fig. 8. Visualisation of results by reprojection on the stereo pair: flies are in pink.

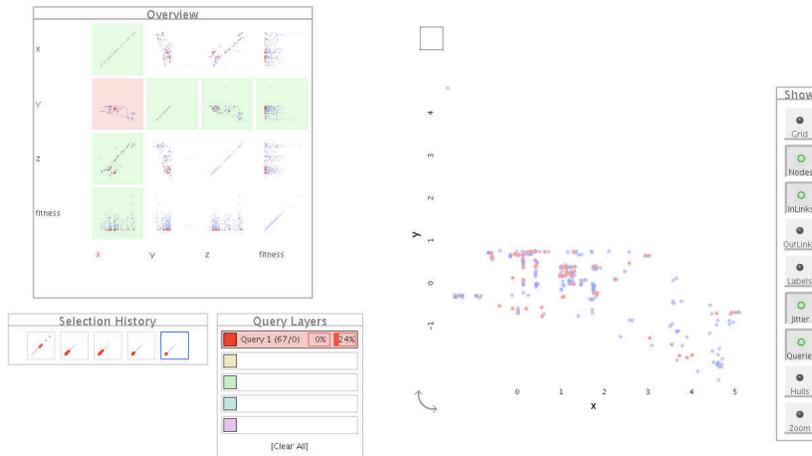


Fig. 9. Front view of the flies cloud: furniture structures are visible. Red points are the best fitted flies.

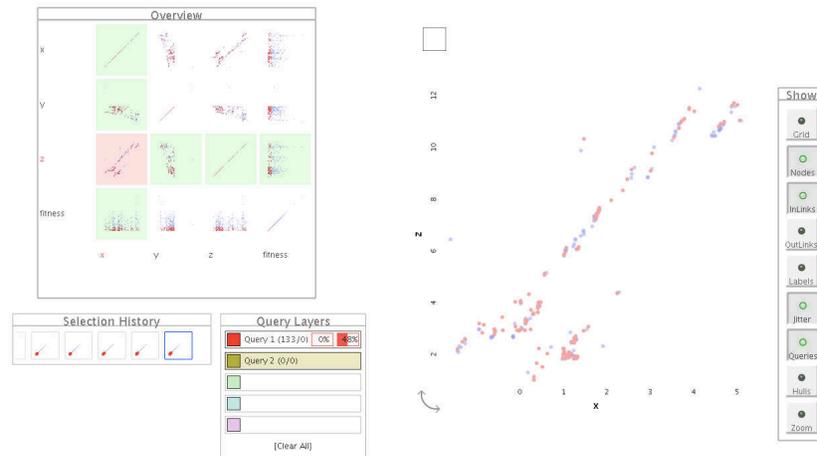


Fig. 10. View from above: the direction of the wall is visible.

6 Conclusion and future work

We have shown above how GraphDice allows fast visual inspection of raw EA data. For instance it is easy to visualise the algorithm’s exploration capability and population diversity, and to localise possible visually obvious dependencies between variables.

Some features of GraphDice are particularly useful to make it a good candidate for a generic answer to the issues identified in section 1: The visualisation scheme is simple and easy to use (2D scatterplots organised in a matrix). Interactions and graphic queries allow a fast navigation in the dataset. Data exchange is performed via simple csv files.

Our intention is now to adapt this general purpose visualisation tool to the specific needs of EA analysis. The following issues will guide future developments of a GraphDice version adapted to EAs:

- Tests have been performed on relatively small data sets (up to 100x100 individuals in 10 dimensional space). The scalability issue will be tested more extensively.
- Various usual statistics (per generation, per fitness level) and query-based statistics will be implemented, including comparison of distributions (p-values).
- On-line visualisation issues will also be considered.

References

1. Mark A. Bedau, Shareen Joshi, and Benjamin Lillie. Visualizing waves of evolutionary activity of alleles. In *Proceedings of the 1999 GECCO Workshop on Evolutionary Computation Visualization*, pages 96–98, 1999.
2. Anastasia Bezerianos, Fanny Chevalier, Pierre Dragicevic, Niklas Elmqvist, and Jean-Daniel Fekete. Graphdice: A system for exploring multivariate social networks. *Computer Graphics Forum (Proc. EuroVis 2010)*, 29(3):863–872, 2010.

3. Seth Bullock and Mark A. Bedau. Exploring the dynamics of adaptation with evolutionary activity plots. *Artif. Life*, 12:193–197, March 2006.
4. P. Collet, E. Lutton, M. Schoenauer, and J. Louchet. Take it EASEA. In M. Schoenauer, K. Deb, G. Rudolf, X. Yao, E. Lutton, Merelo. J.J., and H.-P. Schwefel, editors, *Parallel Problem Solving from Nature - PPSN VI 6th International Conference*, Paris, France, September 16-20 2000. Springer Verlag. LNCS 1917.
5. Trevor D. Collins. *Visualizing evolutionary computation*, pages 95–116. Springer-Verlag New York, Inc., New York, NY, USA, 2003.
6. Andreas Kerren Computer and Andreas Kerren. Eavis: A visualization tool for evolutionary algorithms. In *Proceedings of the IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC 05)*, pages 299–301, 2005.
7. Jason Daida, Adam Hilss, David Ward, and Stephen Long. Visualizing tree structures in genetic programming. *Genetic Programming and Evolvable Machines*, 6:79–110, 2005.
8. Niklas Elmqvist, Pierre Dragicevic, and Jean-Daniel Fekete. Rolling the dice: Multidimensional visual exploration using scatterplot matrix navigation. *IEEE Transactions on Visualization and Computer Graphics (Proc. InfoVis 2008)*, 14(6):1141–1148, 2008.
9. J Foucquier, S Gaucel, C Surel, M Anton, C Garnier, A Riaublanc, C Baudrit, and N Perrot. Modelling the formation of the fat droplets interface during homogenisation in order to describe texture. In *ICEF, 11th International Congress on Engineering and Food, may 22-26, Athens, Greece*, 2011. <http://www.icef11.org/>.
10. Emma Hart and Peter Ross. Gavel - a new tool for genetic algorithm visualization. *IEEE Trans. Evolutionary Computation*, 5(4):335–348, 2001.
11. Evelyne Lutton and Jean-Daniel Fekete. Visual analytics of ea data. In *Genetic and Evolutionary Computation Conference, GECCO 2011*, 2011. July 12-16, 2011, Dublin, Ireland.
12. Z. Mach, M. Zetkova. *Intelligent Technologies - Theory and Applications.*, chapter Visualising genetic algorithms: A way through the Labyrinth of search space., pages 279–285. IOS Press, Amsterdam, 2002. P. Sincak - J. Vascak - V. Kvasnicka - J. Pospichal (eds.).
13. I.C. Parmee and J.A.R. Abraham. Supporting implicit learning via the visualisation of coga multi-objective data. In *CEC2004, Congress on Evolutionary Computation, 19-23 June*, volume 1, pages 395 – 402, 2004.
14. H Pohlheim. Geatbx - genetic and evolutionary algorithm toolbox for matlab. <http://www.geatbx.com/>.
15. Hartmut Pohlheim. Visualization of evolutionary algorithms - set of standard techniques and multidimensional visualization. In *GECCO'99 - Proceedings of the Genetic and Evolutionary Computation Conference, San Francisco, CA*, pages 533–540, 1999.
16. T.W Routen. Techniques for the visualisation of genetic algorithms. In *The First IEEE Conference on Evolutionary Computation*, volume II, pages 846–851, 1994.
17. Emmanuel Sapin, Jean Louchet, and Evelyne Lutton. The fly algorithm revisited: Adaptation to cmos image sensor. In *ICEC 2009, International Conference on Evolutionary Computation*, Madeira, Portugal, October, 5-7 2009.
18. W Shine and C Eick. Visualizing the evolution of genetic algorithm search processes. In *Proceedings of 1997 IEEE International Conference on Evolutionary Computation*, pages 367–372. IEEE Press, 1997.
19. W. M. Spears. An overview of multidimensional visualization techniques. In *Evolutionary Computation Visualization Workshop*, 1999. T. D. Collins, editor, Orlando, Florida, USA.
20. Zbigniew Walczak. Graph-based analysis of evolutionary algorithm. In M Klopotek, S Wierzchon, and K Trojanowski, editors, *Intelligent Information Processing and Web Mining*, volume 31 of *Advances in Soft Computing*, pages 329–338. Springer, 2005.
21. Annie S. Wu, Kenneth A. De Jong, Donald S. Burke, John J. Grefenstette, and Connie Loggia Ramsey. Visual analysis of evolutionary algorithms. In *In Proceedings of the 1999 Conference on Evolutionary Computation (CEC'99)*, pages 1419–1425. IEEE Press, 1999.