Interactive tool for analyzing multiobjective optimization results with Level Diagrams

Xavier Blasco
Instituto de Automática e Informática Industrial.
Universitat Politècnica de València.
Camino de Vera S/N
46022, Valencia, Spain
xblasco@isa.upv.es

Juan Manuel Herrero
Instituto de Automática e Informática Industrial.
Universitat Politècnica de València.
Camino de Vera S/N
46022, Valencia, Spain
juaherdu@isa.upv.es

Gilberto Reynoso-Meza
Industrial and Systems Engineering Graduate Program (PPGEPS).
Pontifical Catholic University of Parana (PUCPR).
Imaculada Conceição, 1155
80215-901, Curitiba, Brasil
g.reynosomeza@pucpr.br

Miguel A. Martínez Iranzo
Instituto de Automática e Informática Industrial.
Universitat Politècnica de València.
Camino de Vera S/N
46022, Valencia, Spain
mmiranzo@isa.upv.es

ABSTRACT

When design problems are multiobjective, graphical representation of the optimization results has a great importance in the analysis and decision making steps. This paper wants to show some of the main features of an interactive tool for multiobjective Pareto front and set analysis. The tool is built in Matlab and uses the *Level Diagram* representation.

It also allows to compare different Pareto fronts and sets. This is especially interesting when the designer wants to compare different kinds of solutions to the same problem. Each type of solution is known as *concept*. For each concept a particular problem of multiobjective optimization is proposed and its Pareto front and set obtained. Although these concepts are usually parametrized in a different decision space, the objectives to be optimized are the same and therefore they can be compared in the same objectives framework.

The interactive tool also supplies the possibility to change colors, shapes and sizes of the points in the representation. These capabilities help the user to understand the relations among the different plots of the Level Diagrams. The norm used in Level Diagrams for axes synchronization offers different points of view about the possible solutions. Therefore, the tool also provides support for easily changing these norms. Level Diagrams with all these extended capabilities are a valuable tool in the decision making process.

GECCO '17 Companion, July 15-19, 2017, Berlin, Germany

ACM ISBN 978-1-4503-4939-0/17/07…$15.00
https://doi.org/http://dx.doi.org/10.1145/3067695.3082553

CCS CONCEPTS

• **Human-centered computing** → **Visualization toolkits**; • **Information systems** → *Data analytics*;

KEYWORDS

Pareto front visualization, Level Diagrams, Graphical interactive tool, Multiobjective decision making

ACM Reference format:
Xavier Blasco, Juan Manuel Herrero, Gilberto Reynoso-Meza, and Miguel A. Martínez Iranzo. 2017. Interactive tool for analyzing multiobjective optimization results with Level Diagrams. In *Proceedings of GECCO '17 Companion, Berlin, Germany, July 15-19, 2017,* 8 pages.
https://doi.org/http://dx.doi.org/10.1145/3067695.3082553

1 INTRODUCTION

Multiobjective optimization [3, 7] deals with the resolution of optimization problems where several conflicting objectives have to be optimized simultaneously. These kinds of problems are common in multiple scientific areas. Without loss of generality, a multiobjective problem is set as:

$$\begin{aligned} &\min_x J(x) \\ \text{subject to:} \quad &x \in S \subset \mathbb{R}^m \end{aligned} \tag{1}$$

where $x = (x_1, \ldots, x_m) \in \mathbb{R}^m$ is defined as the decision vector, $J(x) = (J_1(x), \ldots, J_n(x)) \in \mathbb{R}^n$ as the objective vector, and S as the subspace that satisfies all the additional constraints of the problem.

There is no single solution to this problem because there is no best solution for all objectives. The solution of such problems produces what is called Pareto set as a set of solutions. The definition of this set is based on the concept of Pareto dominance: a decision vector x^1 dominates another vector x^2 if $J(x^1)$ is not worse than $J(x^2)$ in all objectives and is better in at least one objective. Then the Pareto set is defined as the set of all the non-dominated solutions (Pareto optimal solutions) and the Pareto front corresponds to the set of the objective vectors of all Pareto optimal solutions.

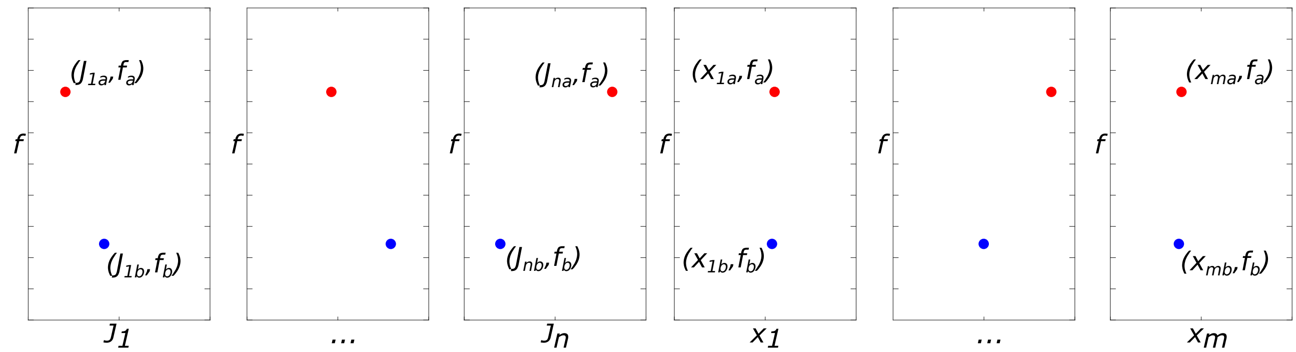

Figure 1: General Level Diagram representation for two Pareto solutions: A and B.

In addition, in engineering problems, the designer can try to solve them through different approaches or design concept. So that, a design concept is an idea on how to solve a particular problem [6]. Each one of these concepts are parametrized in a different decision vector space. The designer will need to compare different concepts in the same framework that is defined by the objective vector space. Then for each concept a multiobjective problem (with the same objectives) has to be set and solved. So that each concept will have its own Pareto set and front.

An adequate resolution of such problems is based on three fundamental steps: a correct selection of the objectives to optimize, a process of optimization where an approximation to the Pareto sets is found and a final decision making process where the designer have to select a preferred solution among the Pareto sets.

For this last step, the designer has to evaluate, analyze and compare several solutions and select the closest to her/his preferences. All kinds of tools to help in this task are welcome. A useful tool should offer the possibility to visualize, in some way, multidimensional data and be able to organize graphical information according to several decision criteria.

Visualization in multiobjective optimization for high dimensional spaces (objective and parameter spaces) is a key issue and the development of new methodologies is attracting more and more interest. [10] and [11] present interesting reviews of some of the existing visualization methods used in multiobjective optimization as well as new alternatives for data visualization. Other recent works show new approaches [4, 5, 10].

The type of information that the decision maker could require from the graphical representation depends on her/his particular procedure for data analysis. But considering that the data sets involved in this step are Pareto front and set approximations, the following characteristics are probably required:

- To evaluate visually the trade-off between objectives in the units of each objective.
- To show the shape of the fronts (convexity, discontinuities, etc.)
- To show dominance relationships.
- To show the position of the Pareto set point according to its position in the Pareto front.
- To compare different Pareto fronts, obtained from different algorithms, or as the result of different design concepts.

Other properties related to data ordering and selection are also necessary:

- Selecting and extracting subsets.
- Coloring points according to designer preferences.
- Changing the size of the points according to designer preferences.
- For Level Diagram representation, the possibility to change the norm for y-axes synchronization.

Specific property for Level Diagram representation:

- The possibility to change the norm for y-axes synchronization.

2 LEVEL DIAGRAM REPRESENTATION

The graphical framework selected in the interactive tool is the *Level Diagram* representation [1, 9]. It has shown good characteristics for multidimensional data analysis with a low computational cost.

2.1 Definition

The level diagram of two related sets of points, for instance a Pareto set and a Pareto front, is a collection of 2D representations synchronized by the y-axis. That is, each coordinate of these points is represented in separated plots where the coordinate of the point is represented in the x-axis and the y-axis is the value of a particular function that shows a property or characteristic of the point. This function commonly uses objective and/or decision vector, but additional information can also be included.

Figure 1 shows a general example with two Pareto solutions (A and B). The space of objectives is n-dimensional, the space of parameters (decision variable) is m-dimensional and the function is 1-dimensional:

$$\mathbf{J} = (J_1, \ldots, J_n) \in \mathbb{R}^n \tag{2}$$

$$\mathbf{x} = (x_1, \ldots, x_m) \in \mathbb{R}^m \tag{3}$$

$$f \in \mathbb{R} \tag{4}$$

$$\begin{array}{lcl} \textit{Solution A} & \rightarrow & (J_{1a}, \ldots, J_{na}),\ (x_{1a}, \ldots, x_{ma}),\ f_a \\ \textit{Solution B} & \rightarrow & (J_{1b}, \ldots, J_{nb}),\ (x_{1b}, \ldots, x_{mb}),\ f_b \end{array}$$

This type of representation has several important characteristics, one of them is that the x-axes are in units of the objectives and parameters respectively. This helps the designer to understand better the range of values attainable and the trade-offs between the different solutions in physical units. A second important characteristic is that the y-axis synchronizes the different plots and supplies a way to show a particular property of each point.

In previous work [1], the y-axis is set to a 2-norm (Euclidean norm) in order to show, in some way, the shape of the multidimensional set. For instance, if the approximated Pareto Front is $\mathbf{J}^*$ (obtained for a Pareto set approximation $\mathbf{x}^*$), to draw the 2-norm level diagram representation, each objective $J_i(\mathbf{x})$, is normalized with respect to its minimum and maximum values. That is:

$$\hat{J}_i(\mathbf{x}) = \frac{J_i(\mathbf{x}) - J_i^{min}}{J_i^{max} - J_i^{min}}, i \in [1 \ldots n]. \tag{5}$$

where:

$$J_i^{min} = \min_{J_i \in J_i^*} J_i \ ; \ J_i^{max} = \max_{J_i \in J_i^*} J_i \tag{6}$$

For each normalized objective vector $\hat{\mathbf{J}}(\mathbf{x}) = (\hat{J}_1(\mathbf{x}), \ldots, \hat{J}_n(\mathbf{x}))$ a 2-norm $f = \|\hat{\mathbf{J}}(\mathbf{x})\|_2 = \sqrt{\sum_{i=1}^{n} \hat{J}_i(\mathbf{x})^2}$ is applied to evaluate the distance to origin in the normalized space and this value is used for y-axis synchronization in level diagram representation.

The choice of f is not limited to Euclidean norm, in fact, many other norms or indicators can be used to show different properties or characteristics of the data. Commonly used norms are:

$$\|\hat{\mathbf{J}}(\mathbf{x})\|_1 = \sum_{i=1}^{n} |\hat{J}_i(\mathbf{x})| \tag{7}$$

$$\|\hat{\mathbf{J}}(\mathbf{x})\|_\infty = \max \hat{J}_i(\mathbf{x}) \tag{8}$$

The 1-norm can help to show convexity properties, whereas the ∞-norm shows better the trade-offs among solutions. However other types of norms can be used. For instance, a composite norm based on an asymmetric norm and a table of preferences is presented in [2]. This norm shows information about preference ranges. A norm to evaluate dominance relationship between to fronts with the quality indicator has been presented in [8, 9]. It allows to visualize the dominance relations between points from different fronts.

3 LEVEL DIAGRAM INTERACTIVE TOOL

The interactive tool[1] has been developed in Matlab and supplies the fundamental interactivity:

- Simultaneous points selections on all 2D plots of a level diagram.
- Representation with several norms.
- Coloring and sizing each point independently.
- Concepts superposition.
- Subsets extractions.

Two types of objects are the base of the tool: *Concept* and *Level Diagram*. The *Concept* object contains the information related to the Pareto front and set in two variables (if the name of the concept is *concept1*, the associated Matlab variables are `concept1` and `concept1_data`). The variable `concept1` is a structure with the following fields:

[1]Available at Matlab file exchange:
https://es.mathworks.com/matlabcentral/fileexchange/62224-interactive-tool-for-multiobjective-optimization-analyze-with-level-diagram-representation

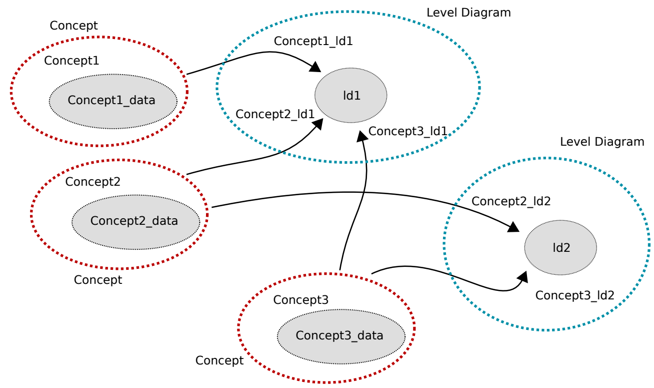

Figure 2: Relations between objects and variables created by the interactive tool. Example with three concepts and two level diagram representation.

- `data`: a string with the name of the variable with Pareto front and set data.
- `nind`: number of solutions in the set.
- `pfdim`: dimension of the Pareto front.
- `psdim`: dimension of the Pareto set.
- `maxpf`: an array with maximum values of the Pareto front.
- `minpf`: an array with minimum values of the Pareto set.
- `maxps`: an array with maximum values of the Pareto front.
- `minps`: an array with minimum values of the Pareto set.

The variable `concept1_data` is a matrix where the first `pfdim` columns store the objectives values and the following `psdim` columns store the parameters values. Each row is a point of the Pareto front and set.

The *Level Diagram* object stores the graphical information and the links to the concepts involved in the graphical representation. It uses one variable for the graphical data (*ldname*). For instance, if a level diagram with the name *ld1* has to be created, the variable `ld1` is created in the Matlab workspace. This variable is a structure with the following fields:

- `concepts`: a *cellarray* with the names (strings) of the concepts represented in this level diagram.
- `figs`: an array with the number of each window used in the graphical representation.
- `axes`: a *cellarray* where each item corresponds to a concept and stores an array with the graphical handlers of each axis used in the graphical representation of this concept.
- `conceptsHandler`: a *cellarray* where each item corresponds to a concept and stores an array with the graphical handlers of the scatter data used for the graphical representation of this concept.

Additionally, for each concept represented in a level diagram, a variable has to be created previously in the Matlab workspace to store the value of the norm used in the level diagram representation (*conceptname_ldname*). This variable is an array with dimension `nind`. For instance, when `ld1` level diagram is created to represent three concepts: `concept1`, `concept2` and `concept3`, three variable `concept1_ld1`, `concept2_ld1` and `concept3_ld1` have to

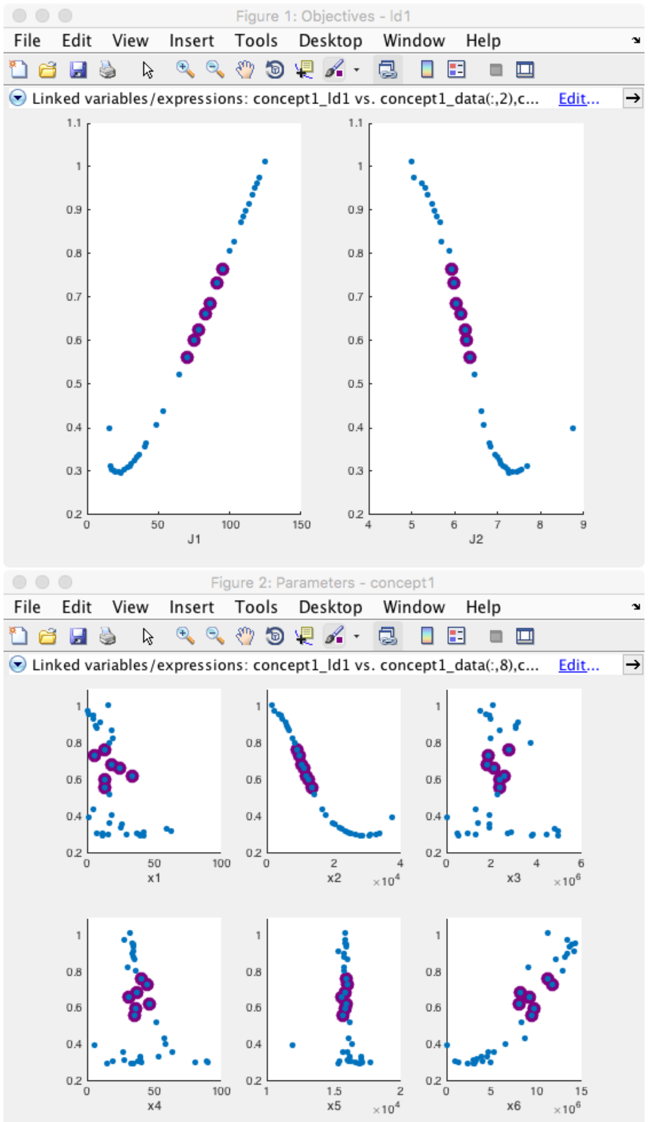

Figure 3: Example of the level diagram windows for a single concept. Some points have been selected with the brush tool of the window.

be created previously with the value of the norm for y-axis synchronization.

Figure 2 shows an example with the relationships of three concepts and two level diagrams. For instance, when the name of the level diagram is *ld2* (the Matlab variable associated is `ld2`), and it contains two concepts (named *concept2* and *concept3*), then the variables to store norm values are `concept2_ld2` and `concept3_ld2`, respectively.

The basic Matlab function developed to maintain the data structure and graphical representation are: `conceptCreate()` to create a concept and `ldDraw()` to draw a level diagram. The way to use these function is described further on. The access to the above mentioned variables supplies a complete control of the graphical representation and data manipulation. A set of functions to increase usability have been created, see Table 1. The following sections shows some of them and several illustrative examples.

Table 1: Main Matlab functions of the interactive tool.

Function	Short description
`conceptCreate()`	Create concept variables from Pareto set and front data.
`ldDraw()`	Draw a concept in a Level Diagram.
`ldChangeMarker()`	Change the marker shape of a concept.
`ldChangeColor()`	Change the color of a concept.
`ldChangeSize()`	Change the size of the marker.
`ldChangeNorm()`	Change the norm for y-axis synchronization.
`ldRefreshAxis()`	Refresh the axis limits of the Level Diagram.

3.1 Coloring and sizing points in a Level Diagram

Several steps are required to draw a level diagram, some variables have to be created previously. First, it is necessary to create two variables associated with a concept (Pareto front and Pareto set approximation). For instance, to create a concept from Pareto front and set data:

```
%% Concept 1
%load Pareto front (pfront6) and set (pset6) data
load('data6.mat')
% Create variables related with the concept:
% concept1 and concept1_data
conceptCreate(pfront6,pset6,'concept1')
```

The second step is to compute the value of the norm/indicator used in the y-axis synchronization and to create the associated variable in the workspace (named with the name of the concept and the level diagram variable name). `basicNorm()` is a function that facilitates the normalization and computation of conventional norms (p-norms), but the user can compute the norm on his own and assign it to the corresponding variable `concept1_ld1`. For instance, if a Euclidean norm is used (with a previous normalization of the front):

```
% Bounds for Pareto front normalization
% bounds =[upperbounds; lowerbounds]
bounds =[124 16; 10 4];% [J1max J1min;J2max J2min]
% Calculate the value of the norm (2-norm) for y-axis ...
    synchronization
% and create the associated variable concept1_ld1
basicNorm('ld1','concept1',bounds,2);
```

Finally, it is necessary to run the following command to draw the level diagram.

```
% Draw the Level Diagram ld1 with a unique concept1
ldDraw('ld1','concept1')
```

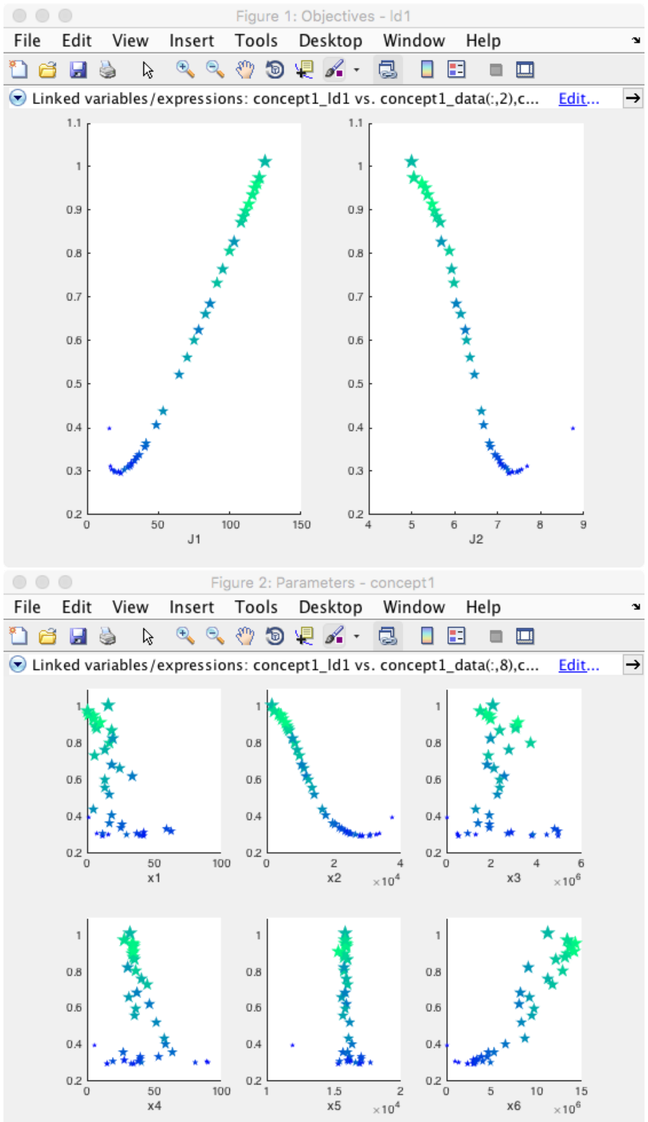

Figure 4: An example of the resulting level diagram when commands for changing marker shape, color and size is applied.

Figure 3 shows the result of running Matlab commands for a basic representation of a level diagram (with 2-norm) and the appearance when some of the points have been selected with the brush tool of the window[2].

Once the level diagram is drawn, it is possible to change the marker shape of the concept and the color and size of the marker for each point accessing the different graphical handlers stored in the level diagram variable. Some commands have been created for this purpose. For instance, for coloring the marker according to the parameter $x6$ value and adjusting their size according to $J2$ value, the following commands are used (Figure 4 shows the result).

```
% The marker shape can be selected between ...
    's','o','^', etc.
% 'p' : pentagram marker
ldChangeMarker(ld1,'p','concept1')
% Creating a RGB matrix to control the color of each ...
    point with the winter colormap of Matlab
c=winter(concept1.nind);
% Creating an array with sizes for each point
s=(20:5:20+5*(concept1.nind-1))';
% Ordering colors according to the values of ...
    parameter x6
[nil,idx]=sort(concept1_data(:,8));
c2(idx,:)=c;
ldChangeColor(ld1,c2,'concept1')
% Ordering sizes according to the values of J1
[nil,idx]=sort(concept1_data(:,1));
s2(idx,1)=s;
ldChangeSize(ld1,s2,'concept1')
```

It is also easy to change the norm for y-axis synchronization. For instance, the following command can be used to replace 2-norm with 1-norm:

```
%% For every concept in ld1, changing the norm  to  ...
    1-norm
ldChangeNorm(ld1,1)
```

3.2 Comparing concepts in a single Level Diagram

Comparing concepts through visualizing is a valuable property of a graphical tool. The developed interactive tool facilitates this task. In general, if the concepts have to be compared in the objective space then it is necessary to superpose the different Pareto fronts. However, when the parameter space is different for each concept, a superposition is not possible. Therefore, when several concepts need to be superposed in the objective space, there is a separated level diagram window for each parameter set.

As an example of use, three concepts are compared using the following commands:

```
%% Load and create Concepts 1, 2 and 3
load('data6.mat')
load('data9.mat')
load('data10.mat')
conceptCreate(pfront6,pset6,'concept1')
conceptCreate(pfront9,pset9,'concept2')
conceptCreate(pfront10,pset10,'concept3')

% Calculate the 2-norm normalizing the front.
% Bounds for Pareto front normalization
bounds =[124 16; 10 4];% [J1max J1min;J2max J2min]
basicNorm('ld1','concept1',bounds,2);
basicNorm('ld1','concept2',bounds,2);
basicNorm('ld1','concept3',bounds,2);

% Draw the Level Diagram ld1 with a unique concept1
ldDraw('ld1','concept1')
ldDraw('ld1','concept2')
ldDraw('ld1','concept3')
```

[2]The Pareto front and set used in these examples (Figures 3, 4 and 5) come from a project with wind turbines. Several controller structures (concepts) with different tuning parameters (obtained from multiobjective optimization) have been compared. The confidentiality clauses of the project prevent further details of the problem.

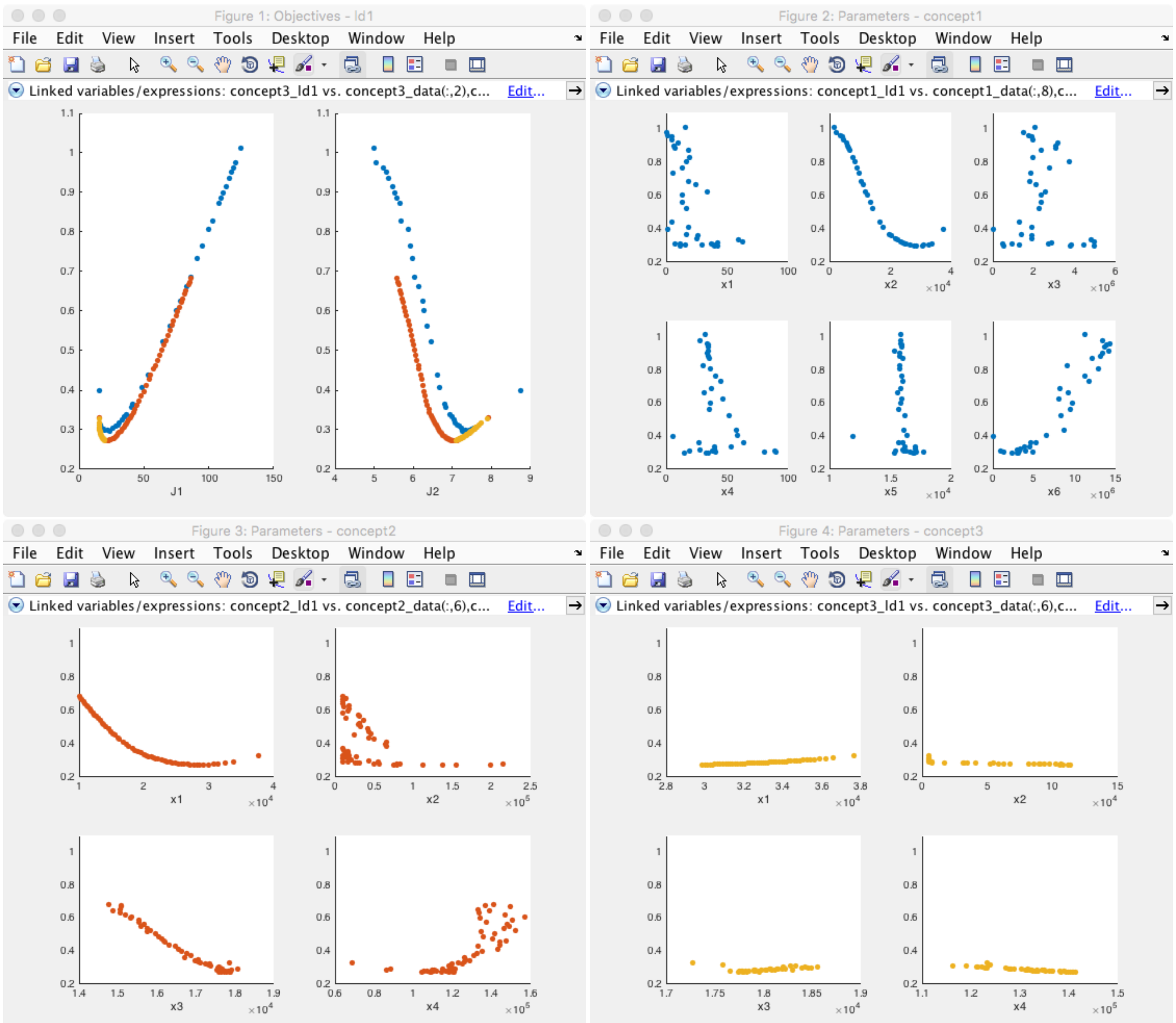

Figure 5: An example of level diagrams representation for concepts superposition.

Figure 5 shows the result of the superposition of the three concepts synchronized by the 2-norm, with a normalization within the same bounds. Remark that the bounds have to be the same for all concepts to ensure that the fronts can be compared.

The labels for x-axes are generic ones: J_1 and J_2 for the objectives and x_1, x_2, etc. for the parameters. The user can change this labels using conventional Matlab commands.

It is also possible to change the marker shape, color and size for each concept individually. For instance, to change the graphic aspect of concept 3:

```
% Change marker shape to hexagram
ldChangeMarker(ld1,'h','concept3')
% Change marker size to 100 pts
ldChangeSize(ld1,100,'concept3')
```

```
% Change marker color to rgb=[0.2 0.6 0.1]
ldChangeColor(ld1,[0.2 0.6 0.1],'concept3')
```

Selection and extraction of a subset of points is done with the brush tool of each window. Once a subset of points is selected, a click with the right mouse button offers the possibility to create a variable in the workspace from this data.

3.3 Comparing concepts in multiple Level Diagrams

Another useful characteristic is that multiple level diagrams can be synchronized offering the possibility to view different types of information. For instance, it can be interesting to visualize the 2-norm to gain insight regarding to the Pareto front shape and the

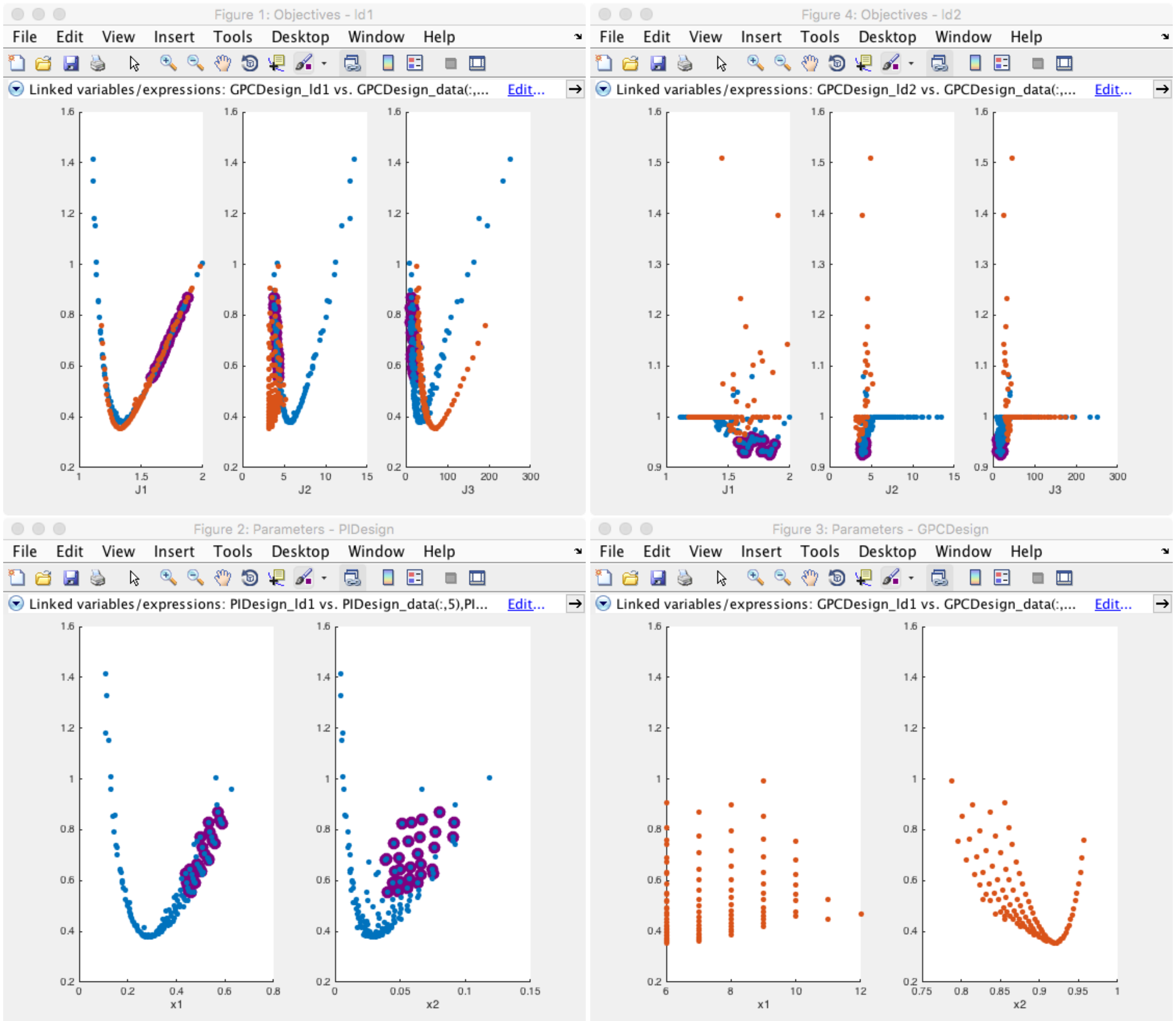

Figure 6: An example of the two level diagrams representation for concepts superposition and comparison. The *Objectives - ld1* (upper left) window shows the Level Diagram for 2-norm and the *Objectives - ld2* (upper rigth) window shows the level diagram with QI. Windows *Parameters - PIDesign* (bottom left) and *Parameters - GPCDesign* (bottom rigth) show the parameters for both concepts.

Quality Indicator (QI) developed in [8] to estimate the dominance relationships among solutions from the two fronts.

In short, given two Pareto front approximations, the QI value is less than 1 for a solution of a given concept if it dominates at least one solution of the other concept; it will be greater than 1 for any solution which is dominated. A QI equal to 1 means that there is not enough information within the Pareto front approximations in order to establish a dominance relation. This last case happens when both fronts coincide or when a front is in a range of values that the other front does not reach (see [8, 9] for more details about QI).

The following example compares two design concepts for a controller design problem, a PI and a GPC controllers, using two level diagrams with two design concepts in each one. In the first level diagram, the 2-norm is used and the Quality Indicator is used in the second one. For more details about this particular problem see [9].

```
% Load data, create concepts, compute 2-norm and draw ld1
load('concepts_data.mat')
conceptCreate(pfrontpi,psetpi,'PIDesign')
conceptCreate(pfrontgpc,psetgpc,'GPCDesign')
bounds=[max([PIDesign.maxpf;GPCDesign.maxpf]);
        min([PIDesign.minpf;GPCDesign.minpf])];
```

```
basicNorm('ld1','PIDesign',bounds,2);
basicNorm('ld1','GPCDesign',bounds,2);
ldDraw('ld1','PIDesign')
ldDraw('ld1','GPCDesign')

% Calculate quality indicator values and create the ...
    variables for concepts PIDesign and GPCDesign ...
    norms in ld2
[PIDesign_ld2,GPCDesign_ld2]=QInorm(pfrontpi,pfrontgpc);
% Draw ld2
ldDraw('ld2','PIDesign')
ldDraw('ld2','GPCDesign')
```

Function `QInorm()` computes the quality indicator for two fronts. Figure 6 shows two level diagrams, where two concepts have been compared with two different indicators (2-norm and QI). The tool offers the possibility to select a subset of points. In Figure 6, some points from concept 1 (PI design) that dominate concept 2 (GPC design) are selected (highlighted). These points have a QI value less than 1 (easy to see in the level diagram 2).

4 CONCLUSIONS

The main characteristics of an interactive tool based on level diagrams have been presented. It has been difficult to show how to interact with the tool in a paper conference. Additional material (screencasts, tutorials and so on) will be made available, and should be read alongside this paper. The tool attempts to supply most of the requirements set by a decision maker to help her/him in the data analysis process. Obviously the requirements are biased by the authors necessities when they face the decision making step, but it seems a good starting point and future improvements are expected. For instance, a front end with menus is already under development in order to facilitate the use of this tool for basic or non Matlab users. Some works about the development of different types of norms/indicators for y-axis synchronization are envisaged.

ACKNOWLEDGMENTS

The work is partially supported by the Spanish Ministry of Economy and Competitiveness program under Grant No.: DPI2015-71443R and by the National Council of Scientific and Technological Development of Brazil under Grant No.: PQ-2/304066/2016-8.

REFERENCES

[1] Xavier Blasco, Juan M. Herrero, Javier Sanchis, and Miguel Martínez. 2008. A new graphical visualization of n-dimensional Pareto front for decision-making in multiobjective optimization. *Information Sciences* 178, 20 (oct 2008), 3908–3924. DOI:http://dx.doi.org/10.1016/j.ins.2008.06.010

[2] Xavier Blasco, Gilberto Reynoso-Meza, Enrique A. Sánchez Pérez, and Juan V. Sánchez Pérez. 2016. Asymmetric distances to improve n-dimensional Pareto fronts graphical analysis. *Information Sciences* 340-341 (may 2016), 228–249. DOI:http://dx.doi.org/10.1016/j.ins.2015.12.039

[3] Carlos A. Coello, David A. Van Veldhuizen, and Gary B. Lamont. 2007. *Evolutionary Algorithms for Solving Multi-Objective Problems.* Springer-Verlag GmbH.

[4] Alan RR de Freitas, Peter J Fleming, and Frederico G Guimarães. 2015. Aggregation Trees for visualization and dimension reduction in many-objective optimization. *Information Sciences* 298 (2015), 288–314.

[5] Amin Ibrahim, Shahryar Rahnamayan, Miguel Vargas Martin, and Kalyanmoy Deb. 2016. 3D-RadVis: Visualization of Pareto front in many-objective optimization. In *2016 IEEE Congress on Evolutionary Computation (CEC).* Institute of Electrical and Electronics Engineers (IEEE). DOI:http://dx.doi.org/10.1109/cec.2016.7743865

[6] Christopher A. Mattson and Achille Messac. 2005. Pareto Frontier Based Concept Selection Under Uncertainty, with Visualization. *Optimization and Engineering* 6, 1 (mar 2005), 85–115. DOI:http://dx.doi.org/10.1023/B:OPTE.0000048538.35456.45

[7] Kaisa Miettinen. 1998. *Nonlinear Multiobjective Optimization.* Springer.

[8] Gilberto Reynoso-Meza, Xavier Blasco, Javier Sanchis, and Juan M. Herrero. 2013. Comparison of design concepts in multi-criteria decision-making using level diagrams. *Information Sciences* 221 (feb 2013), 124–141. DOI:http://dx.doi.org/10.1016/j.ins.2012.09.049

[9] Gilberto Reynoso Meza, Xavier Blasco Ferragud, Javier Sanchis Saez, and Juan Manuel Herrero Durá. 2017. *Controller Tuning with Evolutionary Multiobjective Optimization. A Holistic Multiobjective Optimization Design Procedure.* Springer. DOI:http://dx.doi.org/10.1007/978-3-319-41301-3

[10] Tea Tusar and Bogdan Filipic. 2015. Visualization of Pareto Front Approximations in Evolutionary Multiobjective Optimization: A Critical Review and the Prosection Method. *IEEE Transactions on Evolutionary Computation* 19, 2 (apr 2015), 225–245. DOI:http://dx.doi.org/10.1109/tevc.2014.2313407

[11] David J. Walker, Richard M. Everson, and Jonathan E. Fieldsend. 2013. Visualizing Mutually Nondominating Solution Sets in Many-Objective Optimization. *IEEE Transactions on Evolutionary Computation* 17, 2 (apr 2013), 165–184. DOI:http://dx.doi.org/10.1109/tevc.2012.2225064