Manual of ODEm (Optimal Design Experiments with Matlab)

Ricardo García-Ródenas[1], José Ángel Martín-Baos[1], José Carlos García-García[1]

[1]*Departamento de Matemáticas, Escuela Superior de Informática. Universidad de Castilla-la Mancha. 13071 Ciudad Real, Spain.*

Abstract

ODEm (Optimal Design Experiments with Matlab) is a program developed using Matlab for the computation of optimal design experiments. The program includes heuristic algorithms such as particle swarm optimization (PSO), simulating annealing (SA), genetic algorithm (GA), exact methods such as interior point method (IP), active set method(AS), sequential quadratic programming (SQP), Nelder Mead (NM) and also hybridizations of both exact and heuristic methods.

1. Installation

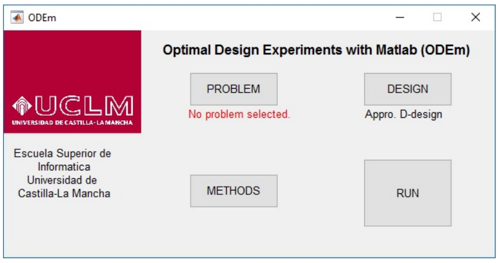
First we are going to explain how to install the program in Windows operating system. The screenshots have been captured in a windows 10 computer, but the program can work in the previous versions (Windows XP, Vista, 7 and 8).

To install the program, the file *ODEm - Windows Installer.exe* must be executed. You only need to follow the steeps given by the installer. If it is the first time you install the application, the MATLAB Runtime library is going to be downloaded from the Internet and installed (about 500 MB). This can take several minutes depending on the quality of the connection. Please be patient.

We recommend you to add a shortcut of the application in your desktop, which can be done clicking the corresponding selection box in the second step. Once the program is installed, you can execute it from the desktop or from the installation directory which by default is *C:\Program Files\UCLM\ODEm*.

It is advisable to read this manual with the ODEm program being executed.

2. Interface


ODEm
UCLM
UNIVERSIDAD DE CASTILLA-LA MANCHA
Escuela Superior de Informatica Universidad de Castilla-La Mancha
Optimal Design Experiments with Matlab (ODEm)
PROBLEM
DESIGN
METHODS
RUN

No problem selected.

Appro. D-design

Figure 1: Screenshot of the main screen of the program.

The program can take aproximately one minute to start because the Matlab modules have to be loaded.

The main screen of the program is composed by 4 buttons, that allow us yo access to each of the sections of the program:

- **Problem.** It allows us to select, to modify or to create a new problem.
- **Design.** It allows us to choose the design of the problem (Approximated or exact design) using a A-, D- or E- criterion.
- **Methods.** It allows to select the different methods that will be used to solve the problem.
- **Run.** It executes the methods selected in 'Methods' on the defined model and shows the results.

Email address: ricardo.garcia@uclm.es (Ricardo García-Ródenas[1], José Ángel Martín-Baos[1], José Carlos García-García[1])

June 16, 2017

3. Example 1: Linear model

In order to understand all the characteristics of the program we are going to work on two examples. In this first example we are going to consider the following linear model: ([2]):
$Y \sim \theta_1 + \theta_2 x_1 + \theta_3 x_1^2 + \theta_4 x_2 + \theta_5 x_1 x_2 + N(0, \sigma^2)$,
$(x_1, x_2) \in [-1, 1] \times [0, 1]$
and $g(\theta) = \theta$

3.1. *Load the problem: Problem button*

This example is already stored in the program as 'Problem2-Linear-two covariates' in the Problem section. We click on the problem name and then in the button SELECT to select the program. We can also modify or check the problem by clicking EDIT or create a new one in CREATE A NEW PROBLEM. In both cases we will access to the problem definition window. Please see Figure 2 for looking how to fill the problem definition window.

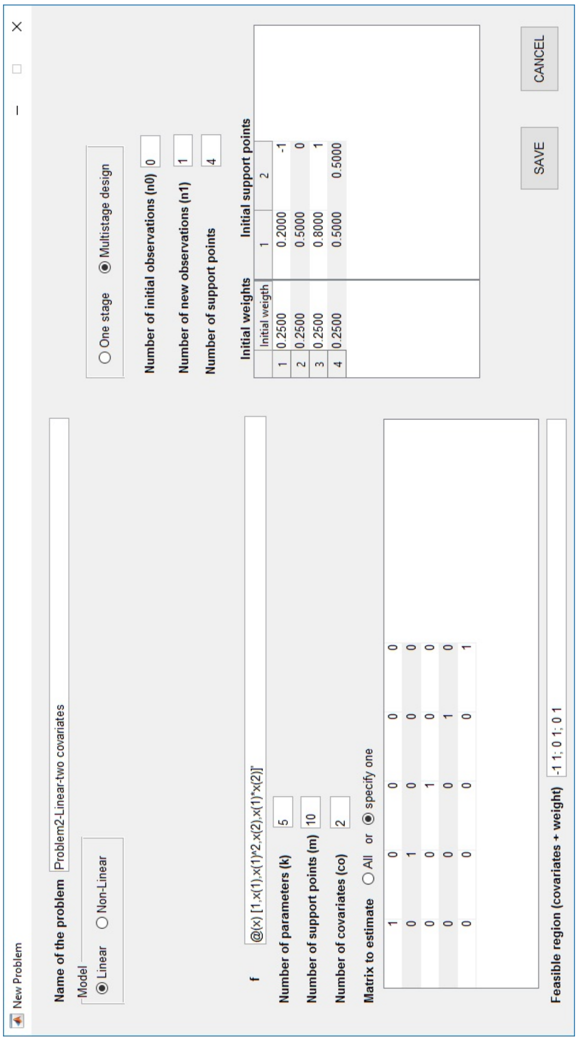
In this problem definition window we can change all the parameters for the problem (such as the number of covariates, parameters, support points, etc...). The function $f(x) = [1, x_1, x_1^2, x_2, x_1 \cdot x_2]^T$ where $Y = f(x)^T \cdot \theta$ is going to be written using Matlab language by means of an anonimous function $@(x)$. Moreover, we can change the limits for the covariates. This field is going to be filled in the following way: We indicate the lower limit and the upper limit of each covariate in order, separating both limits using an space and we will use the semicolon (;) to separate different covariates. At the end, and using a semicolon to separate it, we will indicate the feasible region for the weights [0, 1].

In this example the field is filled in the following way: *-1 1; 0 1; 0 1*. Hence, the feasible region of the first covariate is going to be $[-1, 1]$, the second is going to be $[0, 1]$, and the one for the weight $[0, 1]$.

We can specify also the matrix to estimate. We can use the identity matrix by selecting the `All` option, or we can specify one. In case we want to specify a vector we will do it in the first row, and we are going to fill the rest of the rows with zeros.

It is relevant to indicate that we can select if we are going to use a one stage design or a multistage design here. If we click on multistage design some new options as the prior design and number of experimental subjects are going to appear to configure it.

Back in the problem selection window we will click the problem 2 and then in `Select button`. Then, we will be again in the main window and we will be able to see the problem which is selected under the `Problem button`.



Name of the problem: Problem2-Linear-two covariates

Model: ● Linear ○ Non-Linear

f: @(x) [1,x(1),x(1)^2,x(2),x(1)*x(2)]'

Number of parameters (k): 5

Number of support points (m): 10

Number of covariates (co): 2

Matrix to estimate: ○ All or ● specify one

1	0	0	0	0
0	1	0	0	0
0	0	1	0	0
0	0	0	1	0
0	0	0	0	1

Feasible region (covariates + weight): -1 1; 0 1; 0 1

○ One stage ● Multistage design

Number of initial observations (n0): 0

Number of new observations (n1): 1

Number of support points: 4

	Initial weights	Initial support points	
	Initial weigth	1	2
1	0.2500	0.2000	-1
2	0.2500	0.5000	0
3	0.2500	0.8000	1
4	0.2500	0.5000	0.5000

Figure 2: Screenshot of the problem 2 (Example 1) definition.

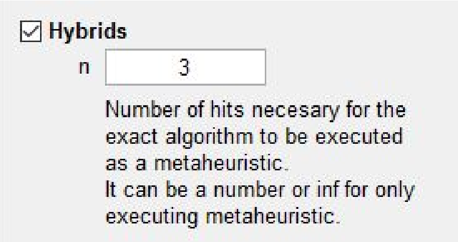
3.2. Design selection: Design button

Now, we are going to choose the `Design`. For doing this, we will access to the Design section. Here, for this example we are going to select an approximated `D-design`: we click on D-design and then on approximated design, and then we click on the `Save` button.

3.3. Methods selection: Methods button

To end with, we are going to select the methods that we will use to solve the problem. We will access to the Methods section. This section consists in a selection panel where you can choose the exacts and metaheuristics algorithms. You can simultaneously choose as many as you want and all of them will be run.

If you select the *Hybrids* option as you can see in Figure 3, an hybrid algorithm will be run. This will combine each one of the selected metaheuristics algorithms with each one of the selected exacts algorithms. The *nc* field specifies the number of descent solutions achieved that will be necessary for the exact algorithm to be run as a metaheuristic.



☑ Hybrids

n 3

Number of hits necesary for the exact algorithm to be executed as a metaheuristic.
It can be a number or inf for only executing metaheuristic.

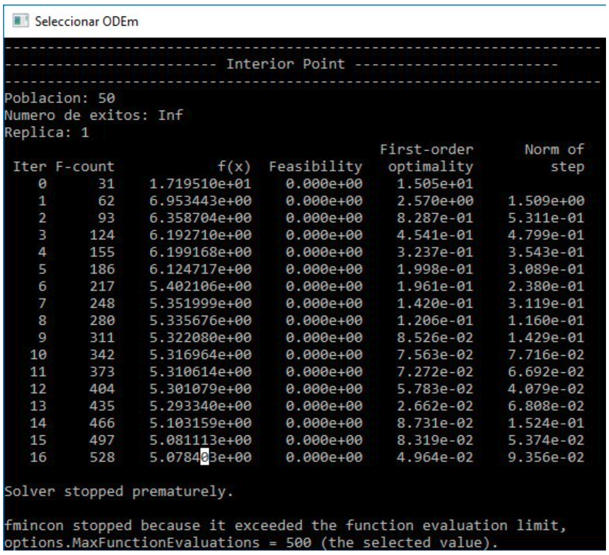
Figure 3: Hybrids option.

Furthermore, you can select the button `More options` for an advanced configuration of the algorithms. These options are: the number of replicas, the number of global iterations, and in the case of hybrids we can also configure the inner iterations for the exact methods and the population used in the algorithms. If the option is not selected, the default values will be used: 1 replica, 500 global iterations, and for the hybrids 100 exacts iterations per execution and a population of 50 individuals. We will change one of these options as an example in the Example 2.

After that we can proceed to the execution.

3.4. Execution of the program: Run button

In this step, we are now going to execute the program. To do this, we only need to click on the `RUN` button. Depending on the selected problem and algorithms, this can take us some seconds or several minutes, please be patient. You can see what is happening in the console which is run at the same time as the program. Look at Figure 4 for looking an example.



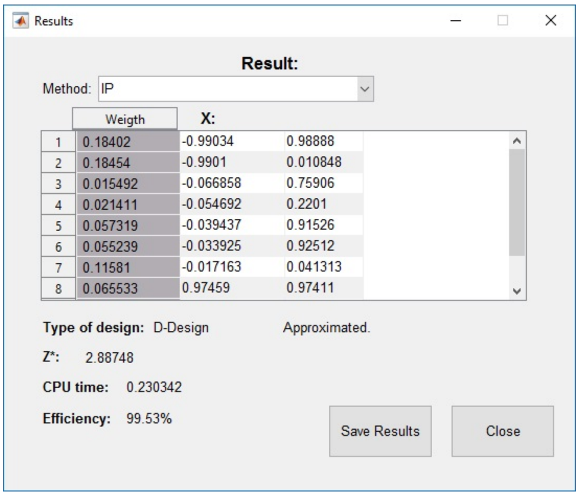
```
----------------------------------------------------------------------
----------------------- Interior Point -----------------------
----------------------------------------------------------------------
Poblacion: 50
Numero de exitos: Inf
Replica: 1
                                              First-order      Norm of
 Iter F-count            f(x)  Feasibility   optimality         step
    0      31    1.719510e+01    0.000e+00    1.505e+01
    1      62    6.953443e+00    0.000e+00    2.570e+00    1.509e+00
    2      93    6.358704e+00    0.000e+00    8.287e-01    5.311e-01
    3     124    6.192710e+00    0.000e+00    4.541e-01    4.799e-01
    4     155    6.199168e+00    0.000e+00    3.237e-01    3.543e-01
    5     186    6.124717e+00    0.000e+00    1.998e-01    3.089e-01
    6     217    5.402106e+00    0.000e+00    1.961e-01    2.380e-01
    7     248    5.351999e+00    0.000e+00    1.420e-01    3.119e-01
    8     280    5.335676e+00    0.000e+00    1.206e-01    1.160e-01
    9     311    5.322080e+00    0.000e+00    8.526e-02    1.429e-01
   10     342    5.316964e+00    0.000e+00    7.563e-02    7.716e-02
   11     373    5.310614e+00    0.000e+00    7.272e-02    6.692e-02
   12     404    5.301079e+00    0.000e+00    5.783e-02    4.079e-02
   13     435    5.293340e+00    0.000e+00    2.662e-02    6.808e-02
   14     466    5.103159e+00    0.000e+00    8.731e-02    1.524e-01
   15     497    5.081113e+00    0.000e+00    8.319e-02    5.374e-02
   16     528    5.078403e+00    0.000e+00    4.964e-02    9.356e-02

Solver stopped prematurely.

fmincon stopped because it exceeded the function evaluation limit,
options.MaxFunctionEvaluations = 500 (the selected value).
```

Figure 4: Program console while executing an Interior Point algorithm.

Now we are going to explain the window which contains the results that will appear when the program has finished executing the computations. In this window we can select what method of the ones selected in the previous section we want to see in detail. You can see the result window in Figure 5.

On the following table of the window we will see the result of the computation. Here we have each result as a row, and the columns are the weigth asociated to each of the covariates which are the rest of the columns under the *X:* label.

Then, we will find some information about the type of design we selected. Besides, we will find the objective function *Z** which is the value of the minimization of the design criterion. This value is the better result obtained by the numerical method. Consequently, the best way to compare the results of several methods is comparing these values and taking the results of the algorithm with the lower one. The computation time taken by the selected algorithm is also reported in *CPU time*. To finish with, we can see the value for the Efficiency of the algorithm.



Result:

Method: IP

	Weigth	X:	
1	0.18402	-0.99034	0.98888
2	0.18454	-0.9901	0.010848
3	0.015492	-0.066858	0.75906
4	0.021411	-0.054692	0.2201
5	0.057319	-0.039437	0.91526
6	0.055239	-0.033925	0.92512
7	0.11581	-0.017163	0.041313
8	0.065533	0.97459	0.97411

Type of design: D-Design Approximated.

Z*: 2.88748

CPU time: 0.230342

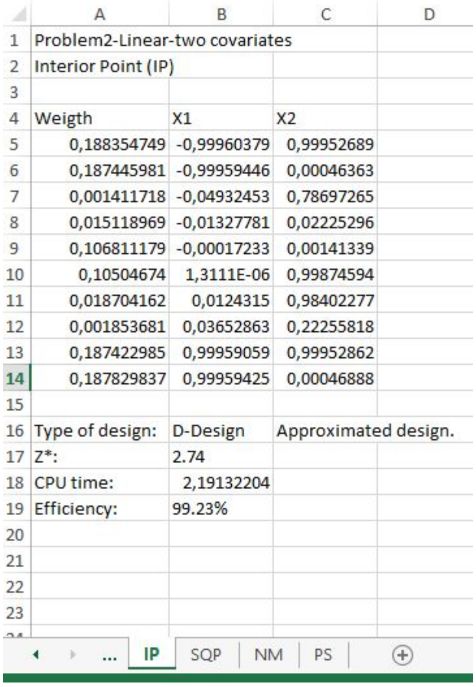
Efficiency: 99.53%

Figure 5: Example 1: Linear model

It is important to have into account some considerations:

- The support points with a weight of 0 have been removed from the solution.
- The results with very close covariates can be considered as one which weight will be the sum of its weight. This consideration is not done automatically by the program. This very similar results are due to the fact that the algorithm has considered it as two or more different support points and has not joined, but in fact they are the same.
- Maybe sometimes the *Efficiency* values are not calculated, represented as "- - -". This is due to its computation requieres to solve an optimization model and the used algorithms may fail.
- In the case of an exact design which will be shown in the second example, each row represents an experimental subject.
- Whenever the method you want to see is changed some minor computations for the Efficiency are required which depending on the problem can take several seconds.

Now we have understood the results, we can save them if we want. To do this we will click on the `Save Results` option. Here we will specify the directory and the name of the excel file when we want to store the data. It is importan that the file where we store the data did not exist previously, if not the data will be overwritten. The process of storing the data can take a while because the excel plug-in used is quite slow, and also because some computations have to be made. Please be patient. In Figure 6 you can see the resulting Excel file.



	A	B	C	D
1	Problem2-Linear-two covariates			
2	Interior Point (IP)			
3				
4	Weigth	X1	X2	
5	0,188354749	-0,99960379	0,99952689	
6	0,187445981	-0,99959446	0,00046363	
7	0,001411718	-0,04932453	0,78697265	
8	0,015118969	-0,01327781	0,02225296	
9	0,106811179	-0,00017233	0,00141339	
10	0,10504674	1,3111E-06	0,99874594	
11	0,018704162	0,0124315	0,98402277	
12	0,001853681	0,03652863	0,22255818	
13	0,187422985	0,99959059	0,99952862	
14	0,187829837	0,99959425	0,00046888	
15				
16	Type of design:	D-Design	Approximated design.	
17	Z*:	2.74		
18	CPU time:	2,19132204		
19	Efficiency:	99.23%		
20				
21				
22				
23				

Figure 6: Example 1. Excel file.

4. Example 2: Non-Linear model

The objective of this example is to show how a general information matrix can be introduced in the ODEm program.

We are going to consider also the following non-linear model: ([2]):

$Y \sim \pi_i(x) = P(Y_i = 1|x) = \frac{e^{h(x)^T\theta_i}}{1+e^{h(x)^T\theta_1}+e^{h(x)^T\theta_2}} i = 1, 2$
$x \in [0, 6]^3$
$h(x) = [1, x^T]^T; g(\theta) = (\theta_1^T, \theta_2^T)^T$
Prior parameters:
$\theta_1 = (1, 1, -1, 2)^T;\quad \theta_2 = (-1, 2, 1, -1)^T$

4.1. Load the problem

This example is already stored in the program as 'Problem3-Multinomial-three covariates' in the Problem section. You can see the problem definition in Figure 7. However we are going to explain it.

First we have to select a Non-linear model and a box will appear to enter auxiliar functions. This will help us to define the information matrix, I_{x_i}. The purpose is to make easier the definition of the model. This auxiliar functions must be written using Matlab language. For more information consult [1].

In this example we have written the following auxiliar functions to define the information matrix:

```
g=@(x) [1,x]'
J=@(x) g(x)*g(x)'
PI=@(x,i) exp(g(x)'*theta_0(:,i))/
  (1+exp(g(x)'*theta_0(:,1))+
  exp(g(x)'*theta_0(:,2)))
```

The above expression define $\Pi_i(x)$ And also we have to define the prior parameter:

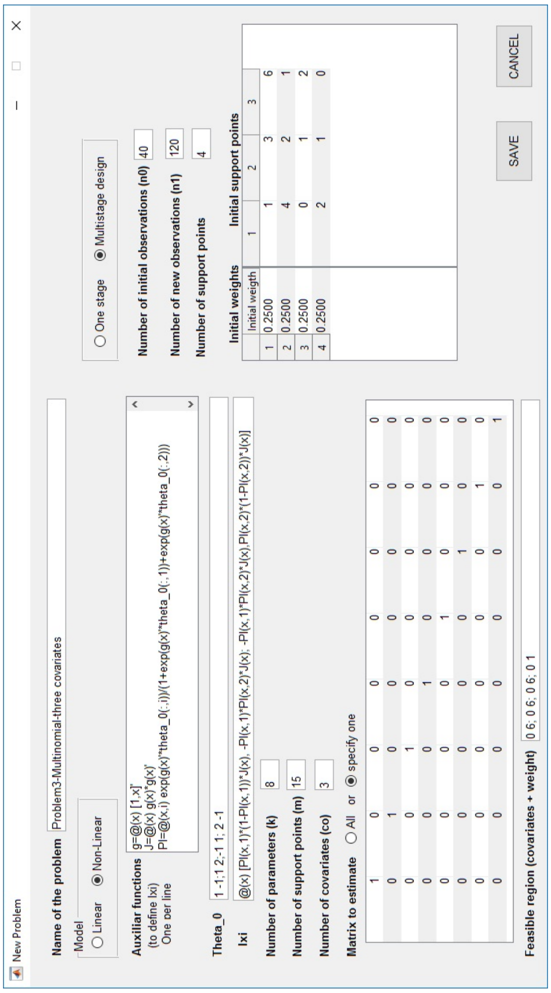
```
Theta_0 as 1 -1;1 2;-1 1; 2 -1
```

This expression means to write the matrix:

$$\begin{array}{cc} \theta_1 & \theta_2 \\ \left[\begin{array}{cc} 1 & -1 \\ 1 & 2 \\ -1 & 1 \\ 2 & -1 \end{array}\right] \end{array}$$

And I_{x_i} becomes:

```
@(x) [PI(x,1)*(1-PI(x,1))*J(x),
  -PI(x,1)*PI(x,2)*J(x);
  -PI(x,1)*PI(x,2)*J(x),
  PI(x,2)*(1-PI(x,2))*J(x)]
```



Name of the problem: Problem3-Multinomial-three covariates

Model: ○ Linear ● Non-Linear

Auxiliar functions (to define Ixi) One per line:

```
g=@(x) [1,x]'
J=@(x) g(x)*g(x)'
PI=@(x,i) exp(g(x)'*theta_0(:,i))/(1+exp(g(x)'*theta_0(:,1))+exp(g(x)'*theta_0(:,2)))
```

Theta_0: 1 -1;1 2;-1 1; 2 -1

Ixi: @(x) [PI(x,1)*(1-PI(x,1))*J(x), -PI(x,1)*PI(x,2)*J(x); -PI(x,1)*PI(x,2)*J(x),PI(x,2)*(1-PI(x,2))*J(x)]

Number of parameters (k): 8

Number of support points (m): 15

Number of covariates (co): 3

Matrix to estimate ○ All or ● specify one

1	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0
0	0	1	0	0	0	0	0
0	0	0	1	0	0	0	0
0	0	0	0	1	0	0	0
0	0	0	0	0	1	0	0
0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	1

Feasible region (covariates + weight): 0 6; 0 6; 0 6; 0 1

○ One stage ● Multistage design

Number of initial observations (n0): 40

Number of new observations (n1): 120

Number of support points: 4

	Initial weights	Initial support points		
	Initial weigth	1	2	3
1	0.2500	1	3	6
2	0.2500	4	2	1
3	0.2500	0	1	2
4	0.2500	2	1	0

Figure 7: Example 2: Non-Linear model

In this example we are going to define a first stage to ilustrate its functionality. We select `Multistage design`, and we will put the number of initial observations to 40. You have to indicate the number of new observations as well as the number of support points, and then we fill the table with the initial support points and their weights.

4.2. *Design selection*

In this example we will use an exact design using 25 individuals. Therefore, in the Design section we will select *Exact design* and put 25 in the *Number of experimental subjects* field.

4.3. *Methods selection*

In this example we are going to select as methods all the exacts: Active-set, Interior-point, Sequential quadratic programming, Nelder-mead and Pattern-search. We can also select some metaheuristics, for example the PSO.

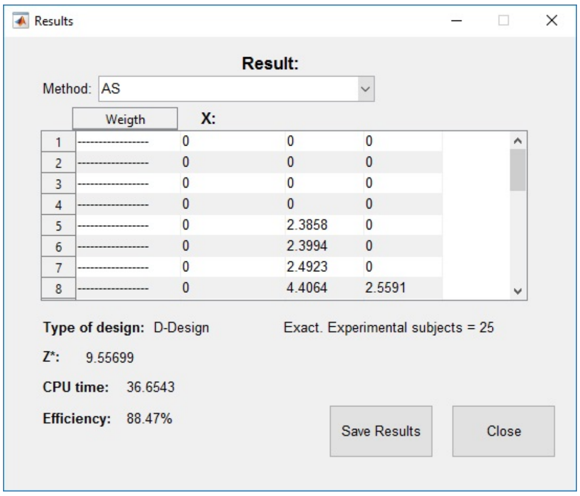
In this example and with the standard configuration we are not going to obtain very good results. Therefore we are going to change the number of global iterations. We click on `More options` and we introduce `5000` global iterations. Now, the algorithms are going to execute more iterations which is going to achieve better solutions. The execution of the algorithms will take lot of time because of the complexity of the problem and the hight number of operations.

4.4. *Execution of the program*

This problem is more complex and it is more difficult to obtain good results. The solution is trying to execute more algorithms or change the configuration as previously mentioned. We could also have tried an hybrid combination.

With the methods selected previously, we are going to look for the one that gives us the best value of Z*: the one that makes it lower. The results can change from execution to execution because the algorithms can take different initial points. For that reason, if you haven't obtained any good result or lots of algorithms has failed, you can rerun the program with the same configuration or try to change it in order to improve the results. For this particular case, we have obtained the best results with the Active-set method, but maybe you can obtain it with another algorithm. The Efficiency is derived from exact designs and it may be poor.

As previously mentioned, an exact design is being used, consequently we will obtain 25 results (rows), one per experimental subject, as you can see in Figure 8.



Result:

Method: AS

	Weigth	X:		
1	----------------	0	0	0
2	----------------	0	0	0
3	----------------	0	0	0
4	----------------	0	0	0
5	----------------	0	2.3858	0
6	----------------	0	2.3994	0
7	----------------	0	2.4923	0
8	----------------	0	4.4064	2.5591

Type of design: D-Design   Exact. Experimental subjects = 25

Z*: 9.55699

CPU time: 36.6543

Efficiency: 88.47%

Figure 8: Example 2: Non-Linear model

5. Using the program in any Operating System

The program can also be used in any other Operating System apart from Windows. In this case you need to install *Matlab* version 2016 or higher and the following Matlab products: *'Statistics and Machine Learning Toolbox'* version 11.0 or higher and *'Global Optimization Toolbox'* version 3.4.1 or higher. Once installed, you need to open the source files of the program and run the *form_main.m* file. It is importat to mention that the `Save Results` option only works on Windows. The matlab files of the ODEm software can be found in: `https://github.com/JoseAngelMartinB/ODEm`

References

[1] Anonymous functions in matlab. `https://es.mathworks.com/help/matlab/matlab_prog/anonymous-functions.html`. Accessed: 2017-01-09.

[2] Yang, M., S. Biedermann, and E. Tang (2013). On optimal designs for nonlinear models: A general and efficient algorithm. Journal of the American Statistical Association 108(504), 1411–1420.