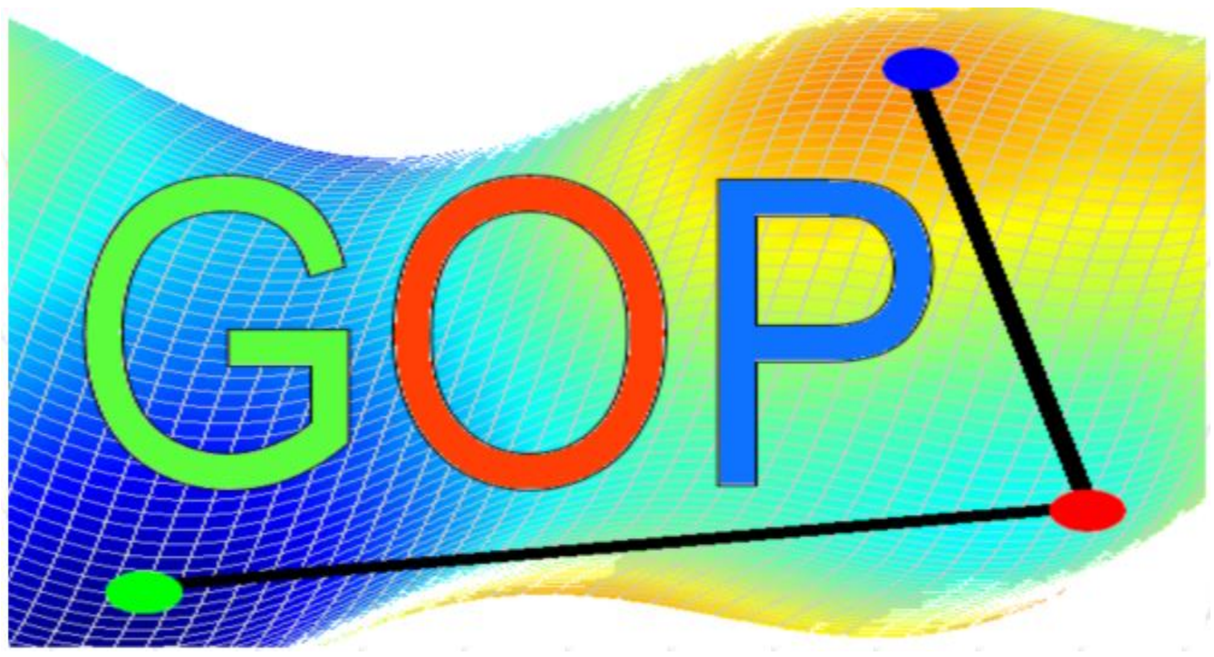


Global Optimization Platform

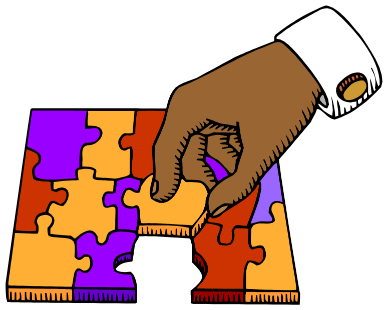
User Guide (Ver. 15.12.11)

Collaborating entities:

License Type:

Licence agreements

Using **Global Optimization Platform** is subject to the acceptance of the following license agreements:



How to contribute to GOP development?

If you are interested to work with us in order to improve the efficiency of the **Global Optimization Platform** you are welcomed.

The philosophy of the **Global Optimization Platform** is to provide a free efficient optimization toolbox to be used in public research programs.

There exist various ways to help us:

- Contributing to the development of the code: Adding core optimization algorithms, developing Toolboxes…
- Testing the robustness of the algorithm: Using this program to solve your own optimisation problem, compare it with other programs and sending us your conclusion and idea of improvement…
- Participating on the debugging of the program: Sending us feedback, error messages, bugs…

Don’t hesitate to contact us.

Sincerely,

- Benjamin Ivorra (Professor at the UCM – Madrid – Spain).
- Bijan Mohammadi (Director of the CERFACS – France).
- Angel Manuel Ramos del Olmo (Professor at the UCM – Madrid – Spain).

What is GOP?

Global Optimization Platform is an optimization software that intends to regroup various class of determinist and stochastic optimisation methods. It allows to solve complex constrained and unconstrained optimization problems. Due to the choice of Matlab, as the development platform, the software is open and easily linkable with many industrial codes (such as COMSOL, Fluent…).

Furthermore, **Global Optimization Platform** includes a new recursive semi-deterministic algorithm [1,2] that intends to improve those methods by optimizing their initial conditions. This approach was first developed at the University Montpellier 2 by the professors Ivorra Benjamin and Bijan Mohammadi. This works is currently continued in collaboration with Professor Angel Manuel Ramos del Olmo and the M.O.M.A.T. research group from the Universidad Complutense de Madrid.

The approaches implemented in **Global Optimization Platform** have been already validated on various industrial and academic optimization problems (see for instance [3,9]). The lecture of associated papers is highly recommended in order to obtain a complete understanding of the method.

You can note that **Global Optimization Platform** is a software in constant evolution, the authors intend to furnish a continuous work in order to improve and debug the program. If you have any suggestion or remark, don't hesitate to contact us.

Table of contents

I. Installation

a. Downloading installation files

You can download installation files at the following direction:
http://www.mat.ucm.es/~ivorra/soft.htm

For the moment there exist four installation versions:

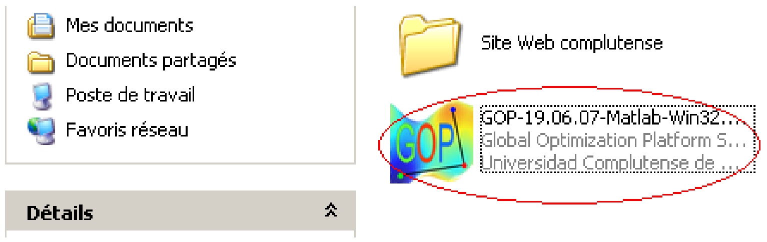
- Windows operating system with Matlab: **GOP-Matlab-Win32.exe**
- All operating systems with Matlab: **GOP-Matlab-All.Zip**
- Windows operating system Stand Alone: **GOP-Stand Alone-Win32.exe**
- Matlab Script (only text I/O): **GOP-Matlab-Script-All.Zip**

Important note: Matlab versions require at least Matlab 7.0 (or newer version) to be installed on your computer. The Stand Alone version requires the MathWorks© **MCRInstaller.exe** to be installed on your computer.

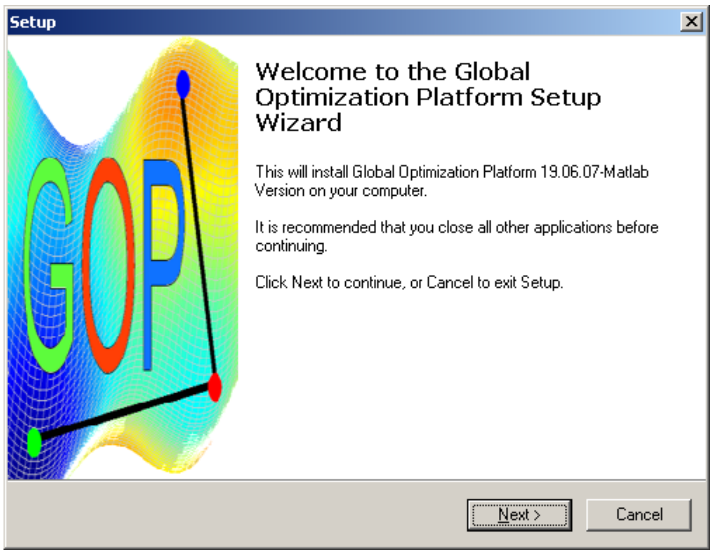
b. Installing the Windows operating system versions

Note for WINDOWS VISTA/7 users: due to security problem, installing GOP in the folders *Program Files* or *Program Files (x86)* could create access errors. We recommend installing GOP in another directory. You can also run GOP in ''*administrator mode*''.

Step 1- Execute **GOP-Matlab-Win32.exe** or **GOP-Stand Alone-Win32.exe** program:


Mes documents
Documents partagés
Poste de travail
Favoris réseau
Détails
Site Web complutense
GOP-19.06.07-Matlab-Win32...
Global Optimization Platform S...
Universidad Complutense de ...

Step 2- The following program presentation window should opens. Click **Next** Button:



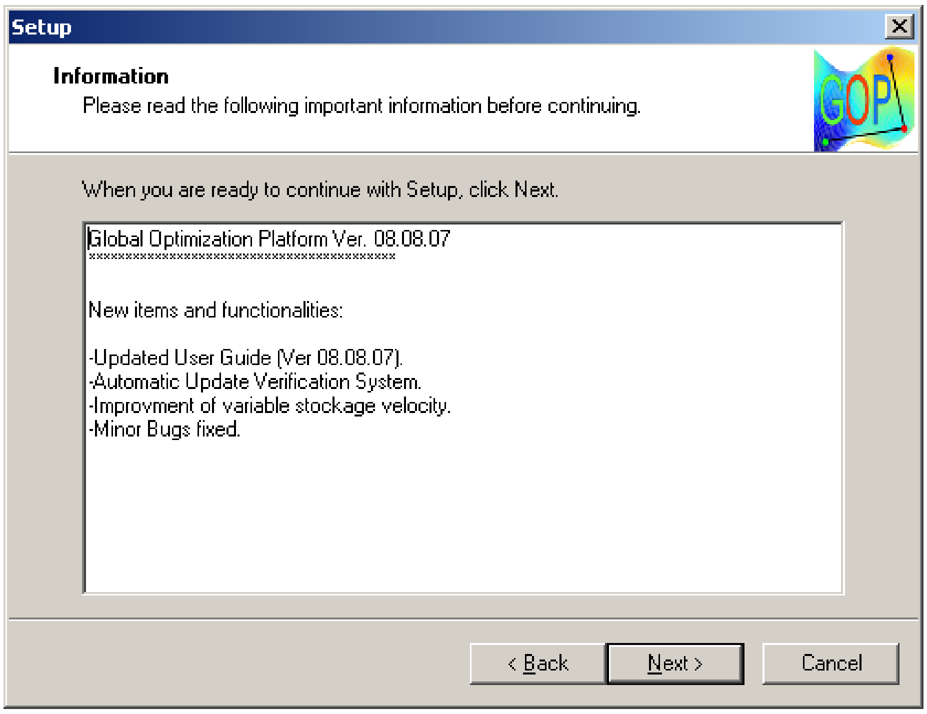
Welcome to the Global Optimization Platform Setup Wizard

This will install Global Optimization Platform 19.06.07-Matlab Version on your computer.

It is recommended that you close all other applications before continuing.

Click Next to continue, or Cancel to exit Setup.

Step 3- Reads the update information and click **Next** Button:



Information

Please read the following important information before continuing.

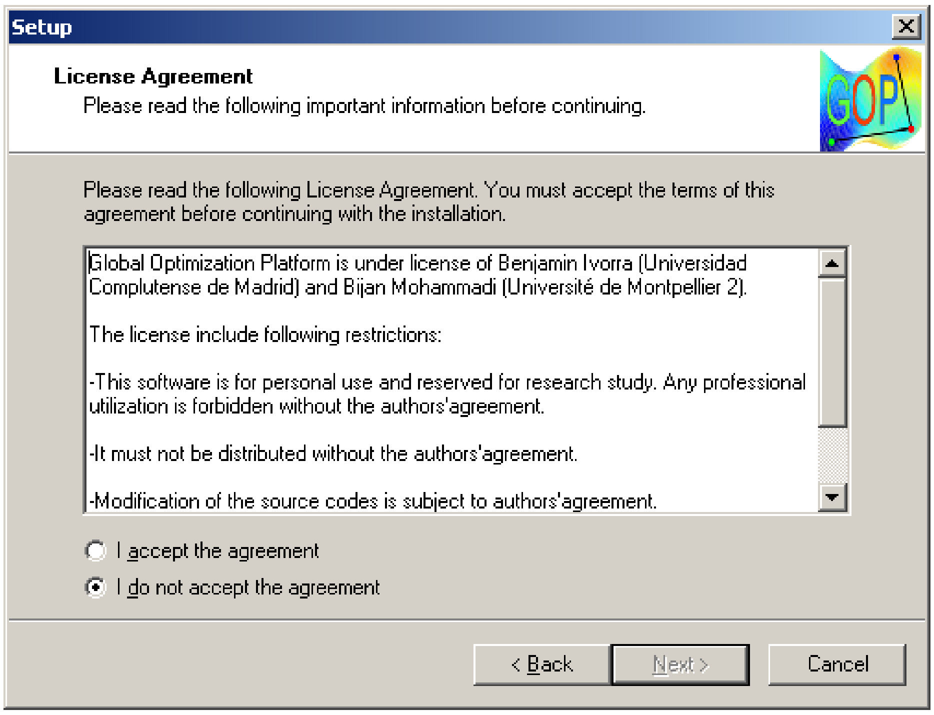
When you are ready to continue with Setup, click Next.

Global Optimization Platform Ver. 08.08.07
****

New items and functionalities:

-Updated User Guide (Ver 08.08.07).
-Automatic Update Verification System.
-Improvment of variable stockage velocity.
-Minor Bugs fixed.

Step 3- Reads the license agreements. If you aggress choose **I accept the agreement** and click **Next** Button:



License Agreement

Please read the following important information before continuing.

Please read the following License Agreement. You must accept the terms of this agreement before continuing with the installation.

Global Optimization Platform is under license of Benjamin Ivorra (Universidad Complutense de Madrid) and Bijan Mohammadi (Université de Montpellier 2).

The license include following restrictions:

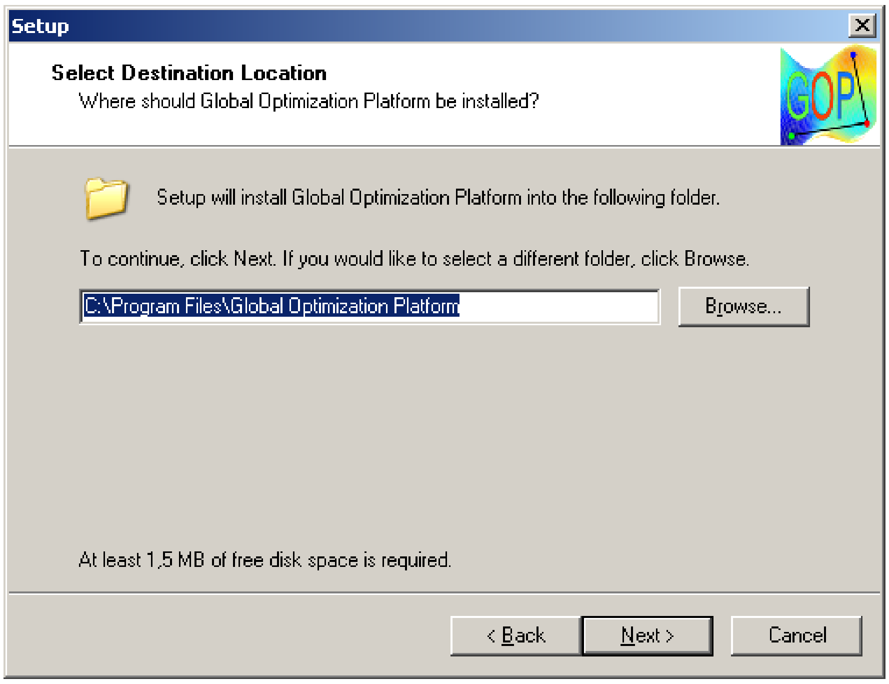
-This software is for personal use and reserved for research study. Any professional utilization is forbidden without the authors'agreement.

-It must not be distributed without the authors'agreement.

-Modification of the source codes is subject to authors'agreement.

I accept the agreement [✓ I do not accept the agreement]

Step 4- Choose the installation directory where GOP will be installed and click **Next** Button:



Select Destination Location
Where should Global Optimization Platform be installed?

Setup will install Global Optimization Platform into the following folder.

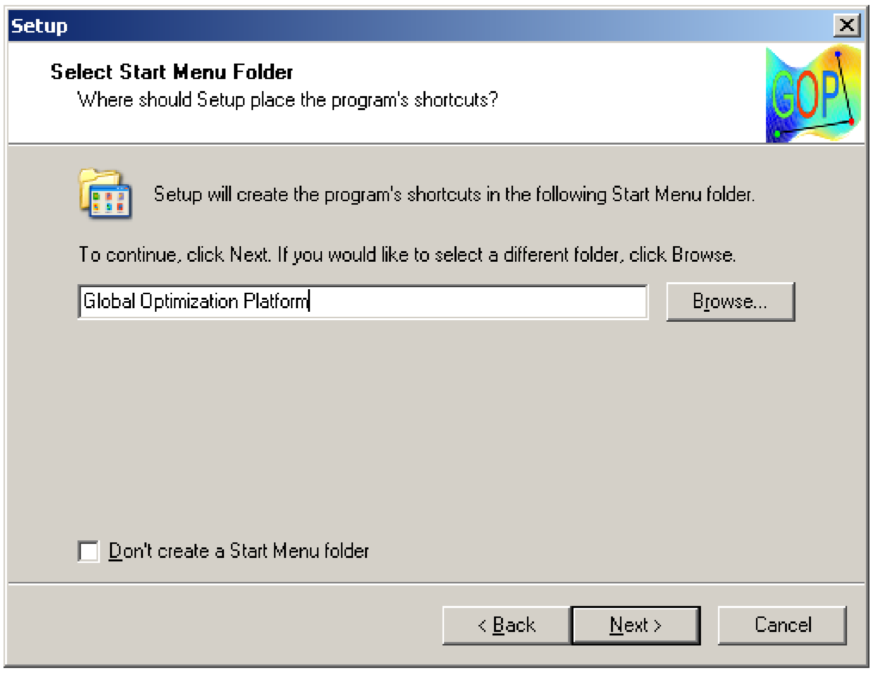
To continue, click Next. If you would like to select a different folder, click Browse.

C:\Program Files\Global Optimization Platform

At least 1,5 MB of free disk space is required.

Note for WINDOWS VISTA/7 users: due to security problem, installing GOP in the folders *Program Files* or *Program Files (x86)* could create access errors. We recommend installing GOP in another directory. You can also run GOP in ''*administrator mode*''.

Step 5- Choose the name of the GOP's Start Menu Folder. If you don't want to create this folder choose the **Don't create a Start Menu folder** option. Click **Next** Button:



Select Start Menu Folder
Where should Setup place the program's shortcuts?

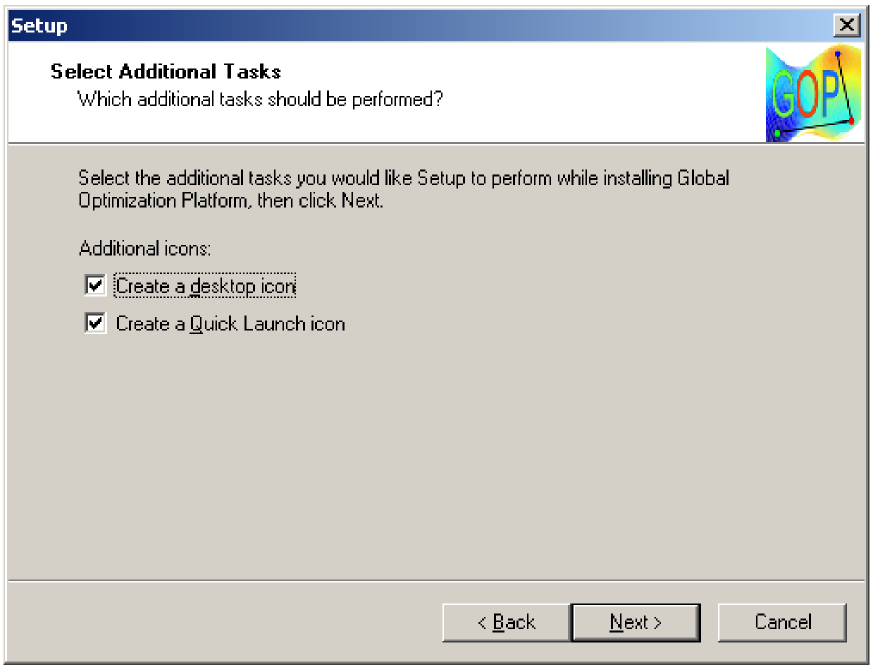
Setup will create the program's shortcuts in the following Start Menu folder.

To continue, click Next. If you would like to select a different folder, click Browse.

Global Optimization Platform

☐ Don't create a Start Menu folder

Step 6- Choose the program icons that should be created. Click **Next** Button:



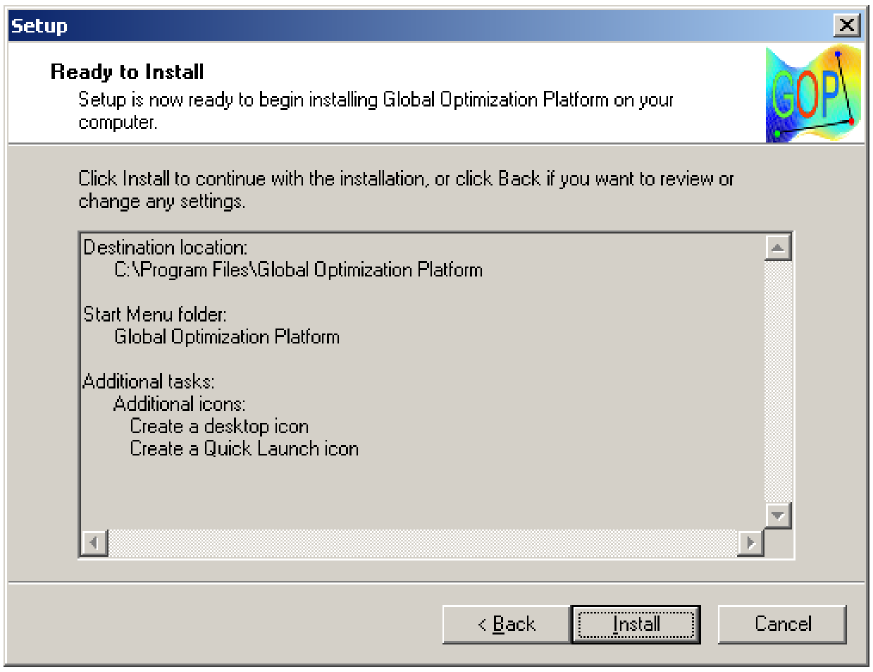
Select the additional tasks you would like Setup to perform while installing Global Optimization Platform, then click Next.

Additional icons:

☑ Create a desktop icon

☑ Create a Quick Launch icon

Step 7- Verify your installation options and click **Install** Button:



Setup is now ready to begin installing Global Optimization Platform on your computer.

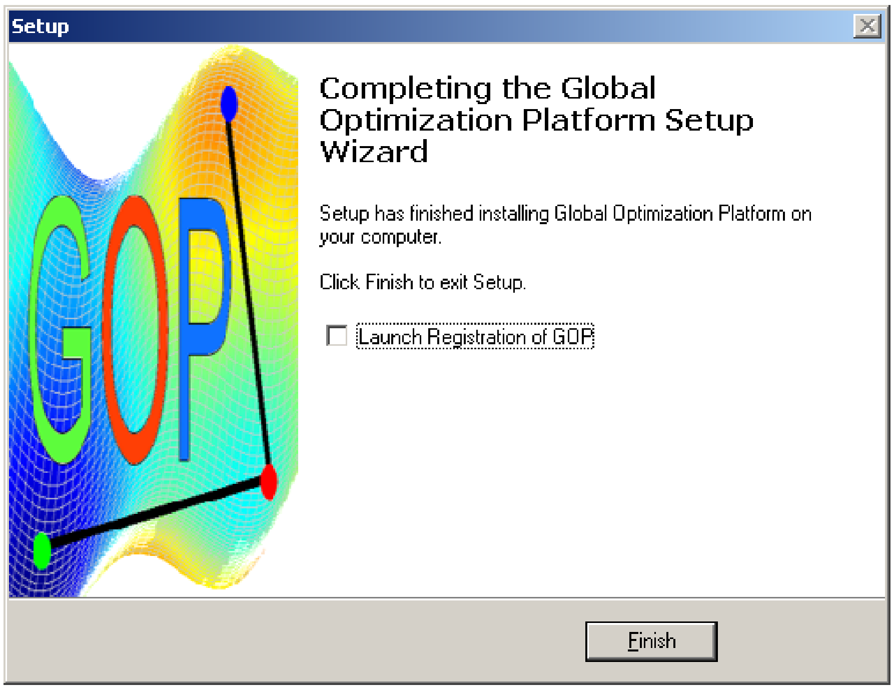
Click Install to continue with the installation, or click Back if you want to review or change any settings.

```
Destination location:
      C:\Program Files\Global Optimization Platform

Start Menu folder:
      Global Optimization Platform

Additional tasks:
      Additional icons:
         Create a desktop icon
         Create a Quick Launch icon
```

Step 8- At the end of the installation process. Choose **Launch Registration of GOP** option if you want to launch GOP and register it immediately. In all cases, the registration process will be launched at the first GOP start. Click on **Finish** button.



Completing the Global Optimization Platform Setup Wizard

Setup has finished installing Global Optimization Platform on your computer.

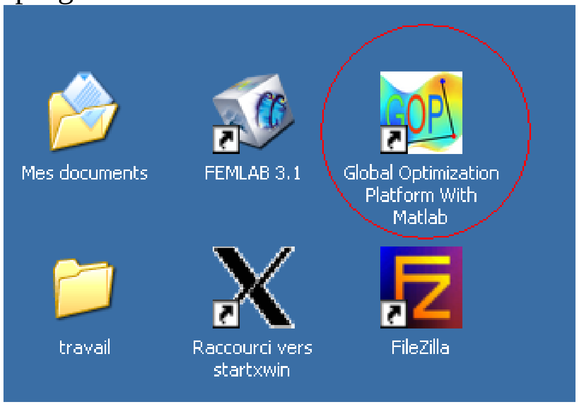
Click Finish to exit Setup.

☐ Launch Registration of GOP

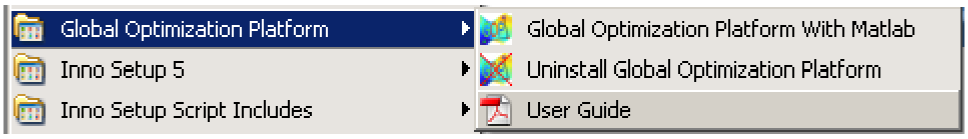
Finish

The **GOP-ver-Matlab Win32.exe** will be closed.

The following **Global Optimization Platform (With Matlab** or **Stand Alone)** Icon should be created on your desktop (if the option has been chosen during installation process). Click on this icon to launch the program:



Furthermore, the following menu should have been created in your Startup Menu (if the option has been chosen during installation process).:

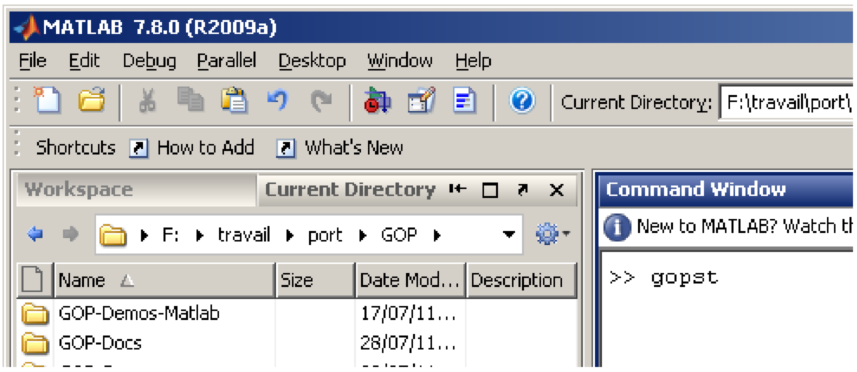

Global Optimization Platform
Inno Setup 5
Inno Setup Script Includes
Global Optimization Platform With Matlab
Uninstall Global Optimization Platform
User Guide

It contains:

- **Global Optimization Platform (With Matlab** or **Stand Alone)** icon that launch the program.
- **Uninstall Global Optimization Platform** icon that uninstall the program.
- **User Guide** icon to read the program instructions.

Step 9- Now you are able to **Launch GOP.** You have two ways to do so:

- Using Windows version: Clicking on the desktop or Startup menu icon **Global Optimization Platform (With Matlab** or **Stand Alone).**
- Using Matlab version: Running Matlab program and executing 'gopst' command in the Matlab command Windows


MATLAB 7.8.0 (R2009a)
File Edit Debug Parallel Desktop Window Help
Current Directory: F:\travail\port\
Shortcuts How to Add What's New
Workspace
Current Directory
Command Window
F: travail port GOP
Name Size Date Mod... Description
New to MATLAB? Watch th

GOP-Demos-Matlab 17/07/11...
GOP-Docs 28/07/11...

```
>> gopst
```

Step 10- The installation process is now finished. Now, you should read the next chapter.

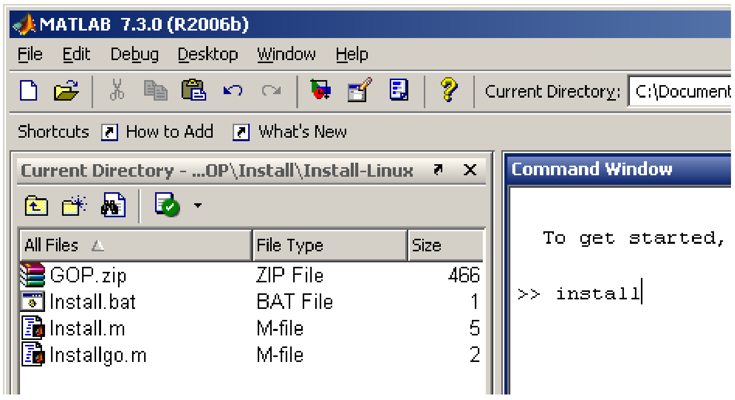
c. Installing the all operating system Matlab version

Step 1- Decompress the archive **GOP-Matlab-All.Zip** inside an empty directory (See your operating system documentation for more information). This directory should now contains the following files:

- **GOP.ZIP**: Contains the program source files.
- **Readme.txt**: Short summary of the installation process.
- **Install.m** and **Installgo.m**: Matlab script performing the program installation.
- **Install.bat**: Batch files launching automatically the previous installation Matlab script.

Step 2- Launch the Matlab installation script using one of those methods:

- Execute **Install.bat** in your operating system shell command.
- Launch Matlab go to the directory used in **Step 1** and execute 'Install' command in the Matlab command Windows

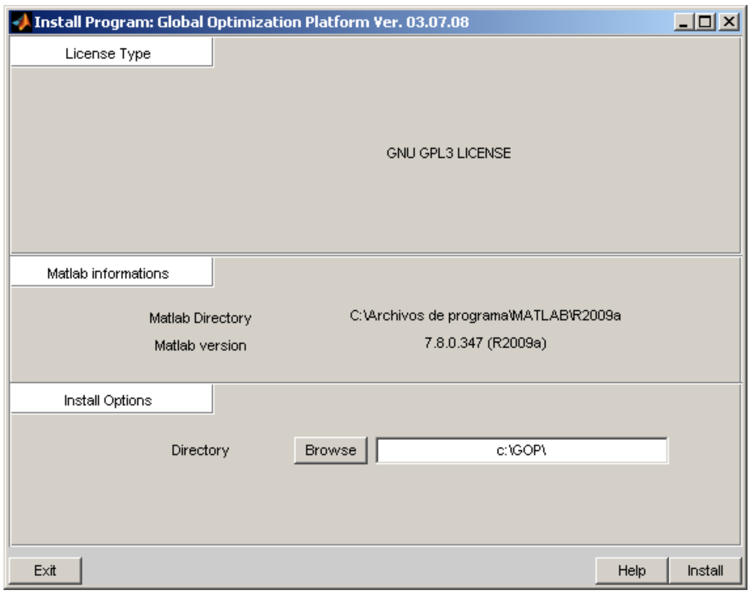

MATLAB 7.3.0 (R2006b)
File Edit Debug Desktop Window Help
Current Directory: C:\Document
Shortcuts How to Add What's New
Current Directory - ...OP\Install\Install-Linux
Command Window

All Files	File Type	Size
GOP.zip	ZIP File	466
Install.bat	BAT File	1
Install.m	M-file	5
Installgo.m	M-file	2

```
To get started,

>> install
```

Step 3- The following Windows should appears:



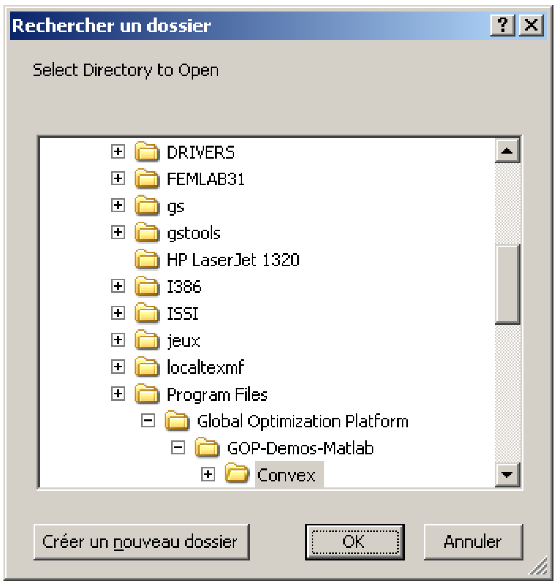
GNU GPL3 LICENSE

Matlab Directory C:\Archivos de programa\MATLAB\R2009a

Matlab version 7.8.0.347 (R2009a)

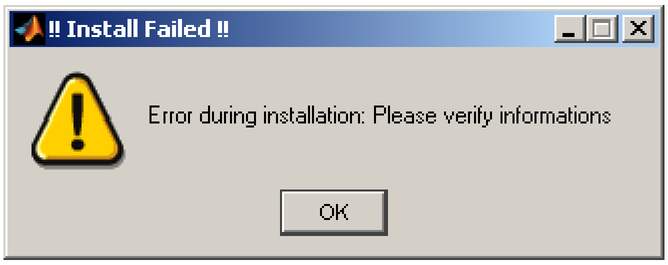
Directory c:\GOP\

Complete the **Install Options** fields: Enter here the directory where you want to install Global Optimization Platform. You could use the browse button I order to navigate, create and choose your installation directory:



Choose the installation directory in and press **OK**. Click **Create a new directory** to create a new directory. Press **Cancel** to stay in the to current directory.

Important note: If the installation directory is incorrect (wrong disk letter,…) the following warning message appears:



!! Install Failed !!

Error during installation: Please verify informations

In this case verify and modify the installation directory.

- The **Exit** button cancel the installation process.
- The **Help** button launch the documentation of the program.

When all fields are completed, click on the **Install** button.

Step 4- The installation process is now finished. Now, you should read the next chapter.

d. Installing the Matlab Script version

In cases when you are using a text I/O system (for example a terminal OS without X11 session), you should use the script version of GOP. By default the script version is included in Matlab versions of GOP, just install GOP as described in previous sections (during this installation you must use a graphical I/O version of Matlab). Then, you can launch the script version executing 'gopscr' command in Matlab (this time you can use a text I/O version).

A only script version can be downloaded independently (**GOP-Matlab-Script.Zip)**. In order to install this version you should follows this process:

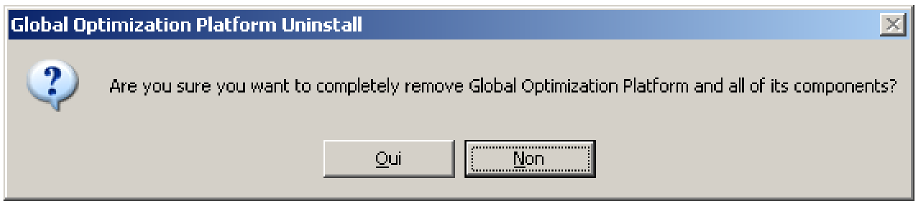
Step 1- Decompress the archive **GOP-Matlab-Script.Zip** inside an empty directory (See your operating system documentation for more information).

Step 2- In the same directory run Matlab and enter 'gopscr' command.

e. Uninstalling Global Optimization Platform

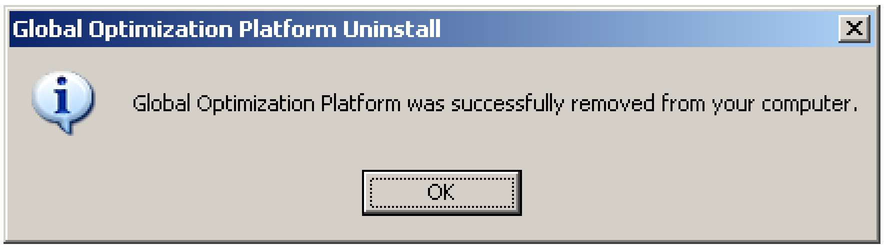
There exist two ways to uninstall this program:

- Windows versions: Use the Uninstall Startup Menu Icon. TH following Windows is displayed:


Global Optimization Platform Uninstall

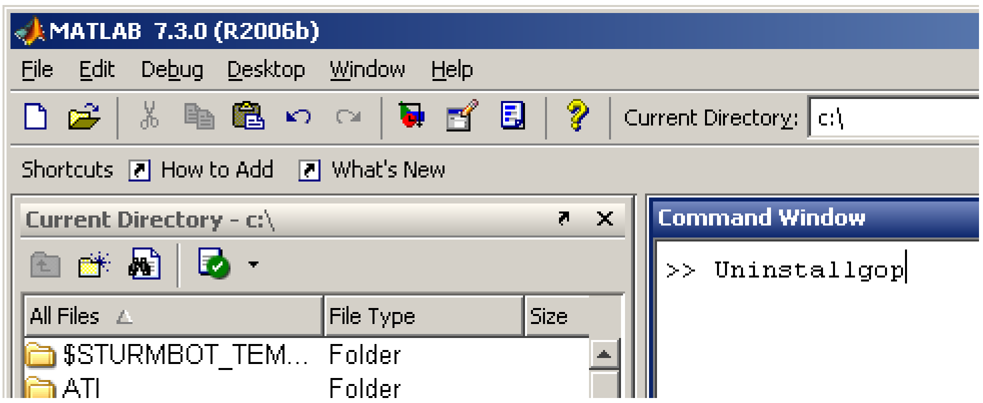
Are you sure you want to completely remove Global Optimization Platform and all of its components?

Select **No** to cancel the uninstallation or **Yes** to confirm the uninstallation. The uninstallation is then launched automatically. At the end of this process, the following message appears:


Global Optimization Platform Uninstall

Global Optimization Platform was successfully removed from your computer.

- All operating system Matlab version: Launch Matlab and execute 'Uninstallgop' command in the Matlab command Windows



```
>> Uninstallgop
```

Important note: In both cases you should verify that the install directory has been removed. In fact due to the fact you should have worked on this directory, some personal files may have been kept inside this directory. In order to avoid to loose important files, only a manual suppression is allowed.

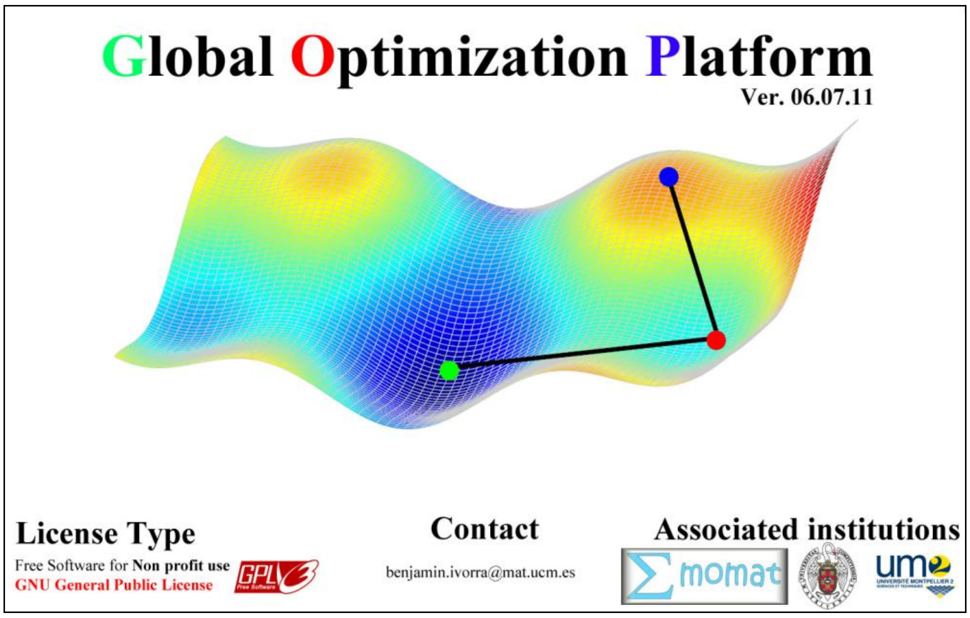
f. Launching GOP

Depending on the version installed GOP can be launched using two ways:

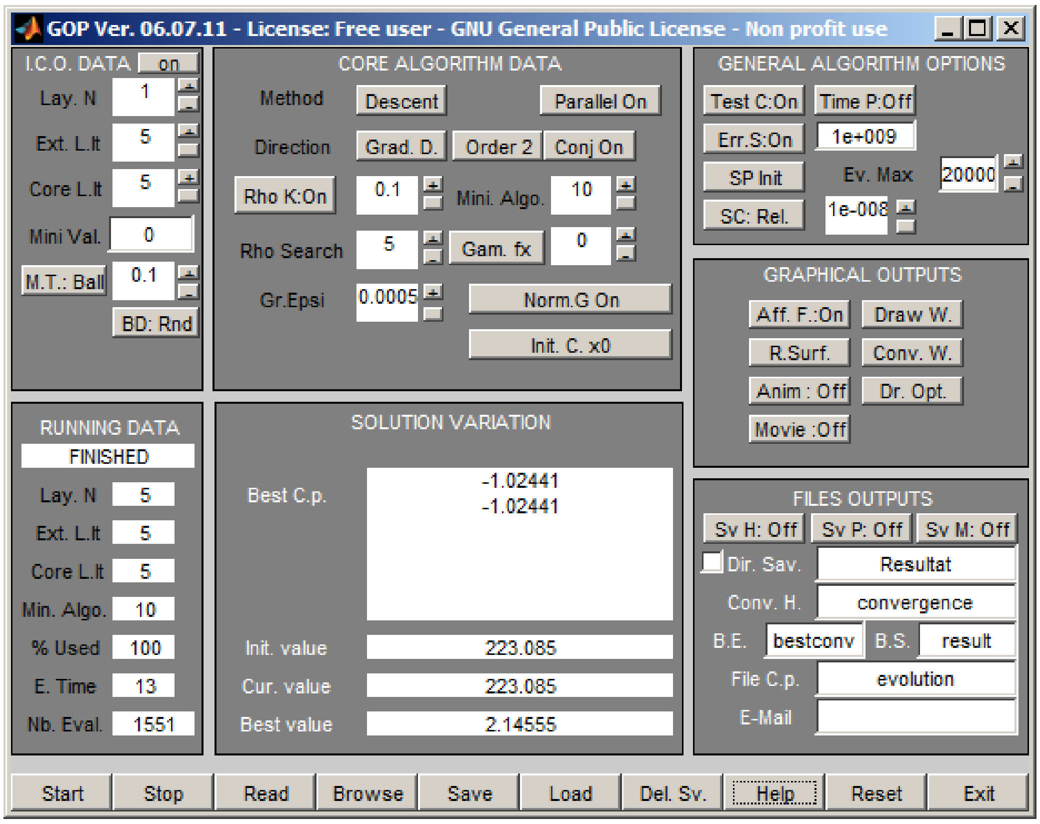
- Using Windows versions: Clicking on the desktop or Startup menu icon.
- Using Matlab versions: Running Matlab program and executing 'gopst' command in the Matlab command window.

- <u>Using Matlab Script version:</u> Running Matlab program and executing ‘gopscr’ in the Matlab command window.

The following title Windows should appears:


Global Optimization Platform
Ver. 06.07.11
License Type
Free Software for Non profit use
GNU General Public License
GPLv3
Contact
benjamin.ivorra@mat.ucm.es
Associated institutions
momat
um2

Click or push any key to pass this image. GOP main Windows is now displayed:


GOP Ver. 06.07.11 - License: Free user - GNU General Public License - Non profit use

I.C.O. DATA: on
Lay. N: 1
Ext. L.It: 5
Core L.It: 5
Mini Val.: 0
M.T.: Ball: 0.1
BD: Rnd

CORE ALGORITHM DATA
Method: Descent — Parallel On
Direction: Grad. D. — Order 2 — Conj On
Rho K:On: 0.1 — Mini. Algo.: 10
Rho Search: 5 — Gam. fx: 0
Gr.Epsi: 0.0005 — Norm.G On
Init. C. x0

GENERAL ALGORITHM OPTIONS
Test C:On — Time P:Off
Err.S:On: 1e+009
SP Init — Ev. Max: 20000
SC: Rel.: 1e-008

GRAPHICAL OUTPUTS
Aff. F.:On — Draw W.
R.Surf. — Conv. W.
Anim : Off — Dr. Opt.
Movie :Off

RUNNING DATA
FINISHED
Lay. N: 5
Ext. L.It: 5
Core L.It: 5
Min. Algo.: 10
% Used: 100
E. Time: 13
Nb. Eval.: 1551

SOLUTION VARIATION
Best C.p.: -1.02441 -1.02441
Init. value: 223.085
Cur. value: 223.085
Best value: 2.14555

FILES OUTPUTS
Sv H: Off — Sv P: Off — Sv M: Off
Dir. Sav.: Resultat
Conv. H.: convergence
B.E.: bestconv — B.S.: result
File C.p.: evolution
E-Mail:

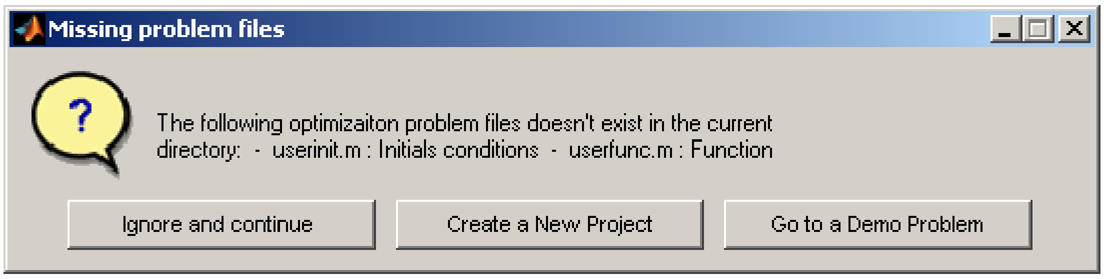
Start — Stop — Read — Browse — Save — Load — Del. Sv. — Help — Reset — Exit

Note for the text I/O versions: The following text should appears:

```
GOP Ver. 08.07.07 Initilization successful

Welcome to the script version of GOP
```

Important note: If it's the first time that you launch GOP or no **optimisation problems files** (see **Chapter IV b**) are present in the working directory, the following message should appear:


Missing problem files

The following optimizaiton problem files doesn't exist in the current directory: - userinit.m : Initials conditions - userfunc.m : Function

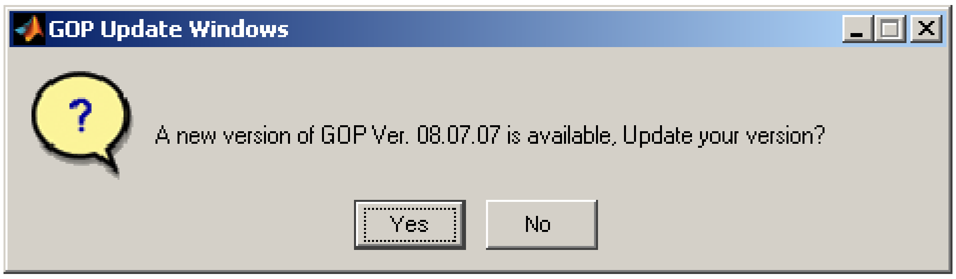
We will describe in **Chapter V** those three options.

Note for the text I/O versions: The following text should appears:

```
Warning: The following files doesn't exist :  -  userinit.m : Initials conditions
-  userfunc.m : Function
```

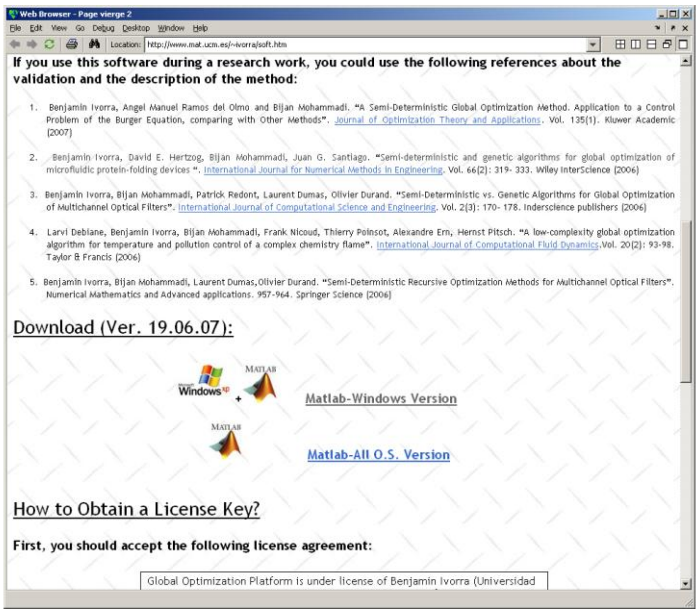
g. Automatic Update

When your computer is connected to internet , GOP automatically verify if a new version of GOP is available. If he find a new version the following message appears:


GOP Update Windows

A new version of GOP Ver. 08.07.07 is available, Update your version?

If you click **No**, GOP continues normally. If you click **Yes** GOP is stopped and a Web-browser open at the GOP website. You should download and install the new version:


Web Browser - Page vierge 2
File Edit View Go Debug Desktop Window Help
Location: http://www.mat.ucm.es/~ivorra/soft.htm

If you use this software during a research work, you could use the following references about the validation and the description of the method:

1. Benjamin Ivorra, Angel Manuel Ramos del Olmo and Bijan Mohammadi. "A Semi-Deterministic Global Optimization Method. Application to a Control Problem of the Burger Equation, comparing with Other Methods". Journal of Optimization Theory and Applications. Vol. 135(1). Kluwer Academic (2007)
2. Benjamin Ivorra, David E. Hertzog, Bijan Mohammadi, Juan G. Santiago. "Semi-deterministic and genetic algorithms for global optimization of microfluidic protein-folding devices ". International Journal for Numerical Methods in Engineering. Vol. 66(2): 319- 333. Wiley InterScience (2006)
3. Benjamin Ivorra, Bijan Mohammadi, Patrick Redont, Laurent Dumas, Olivier Durand. "Semi-Deterministic vs. Genetic Algorithms for Global Optimization of Multichannel Optical Filters". International Journal of Computational Science and Engineering. Vol. 2(3): 170- 178. Inderscience publishers (2006)
4. Larvi Debiane, Benjamin Ivorra, Bijan Mohammadi, Frank Nicoud, Thierry Poinsot, Alexandre Ern, Hernst Pitsch. "A low-complexity global optimization algorithm for temperature and pollution control of a complex chemistry flame". International Journal of Computational Fluid Dynamics.Vol. 20(2): 93-98. Taylor & Francis (2006)
5. Benjamin Ivorra, Bijan Mohammadi, Laurent Dumas,Olivier Durand. "Semi-Deterministic Recursive Optimization Methods for Multichannel Optical Filters". Numerical Mathematics and Advanced applications. 957-964. Springer Science (2006)

Download (Ver. 19.06.07):

Matlab-Windows Version

Matlab-All O.S. Version

How to Obtain a License Key?

First, you should accept the following license agreement:

Global Optimization Platform is under license of Benjamin Ivorra (Universidad

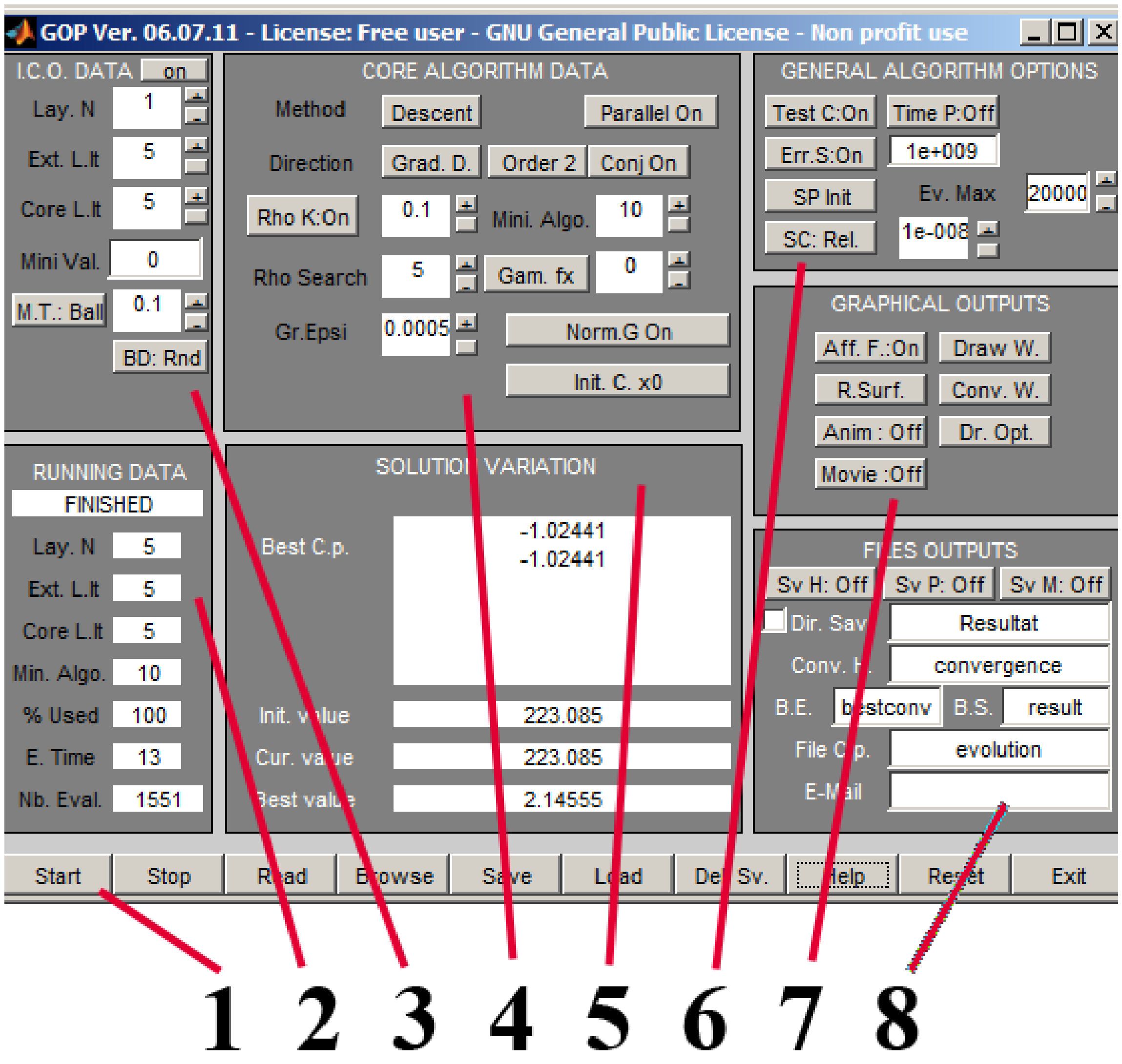
Note for the text I/O versions: The following text should appears:

```
Warning: A new version of GOP Ver. 08.07.07 is available, you should update your
version at http://www.mat.ucm.es/~ivorra/soft.htm
```

II. Graphical User Interface and Options

a. GOP main Windows

The main Windows may be decomposed by 8 zones:



GOP Ver. 06.07.11 - License: Free user - GNU General Public License - Non profit use

I.C.O. DATA: on
Lay. N: 1
Ext. L.It: 5
Core L.It: 5
Mini Val.: 0
M.T.: Ball: 0.1
BD: Rnd

CORE ALGORITHM DATA
Method: Descent
Parallel On
Direction: Grad. D. | Order 2 | Conj On
Rho K:On: 0.1
Mini. Algo.: 10
Rho Search: 5
Gam. fx: 0
Gr.Epsi: 0.0005
Norm.G On
Init. C. x0

GENERAL ALGORITHM OPTIONS
Test C:On | Time P:Off
Err.S:On: 1e+009
SP Init
Ev. Max: 20000
SC: Rel.: 1e-008

GRAPHICAL OUTPUTS
Aff. F.:On | Draw W.
R.Surf. | Conv. W.
Anim : Off | Dr. Opt.
Movie :Off

RUNNING DATA
FINISHED
Lay. N: 5
Ext. L.It: 5
Core L.It: 5
Min. Algo.: 10
% Used: 100
E. Time: 13
Nb. Eval.: 1551

SOLUTION VARIATION
Best C.p.: -1.02441 -1.02441
Init. value: 223.085
Cur. value: 223.085
Best value: 2.14555

FILES OUTPUTS
Sv H: Off | Sv P: Off | Sv M: Off
Dir. Sav: Resultat
Conv. H: convergence
B.E.: bestconv
B.S.: result
File C.p.: evolution
E-Mail:

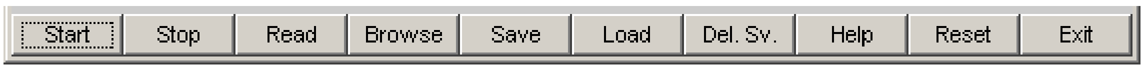
Start | Stop | Read | Browse | Save | Load | Del Sv. | Help | Reset | Exit

1. Menu Bar
2. Running Data
3. I.C.O Data
4. Core Algorithm Data
5. Solution Variation
6. General Algorithm Options
7. Graphical Outputs
8. Files Outputs

We will describe step by step the use and options of each zone.

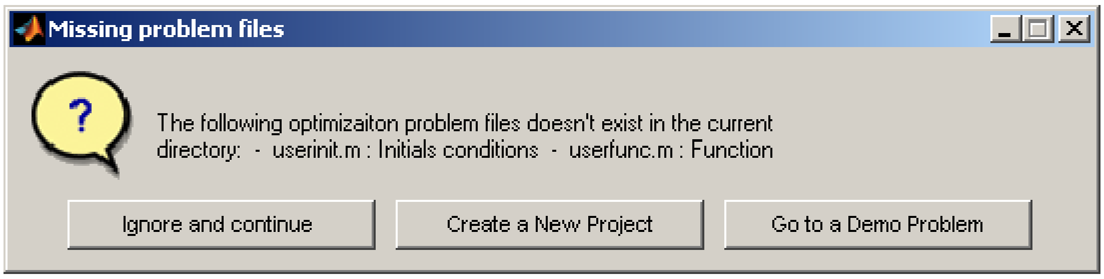
b. Zone 1- Menu Bar

Menu with the principal control commands:



- **Start**: Launch optimization algorithm.

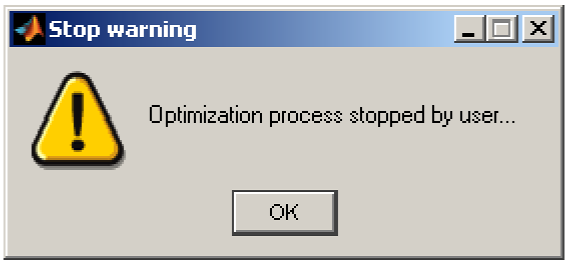
Important note: If **userinit.m** or **userfunc.m** files are missing (for example you are not in the good directory) the following warning message is displayed:


Missing problem files

The following optimizaiton problem files doesn't exist in the current directory: - userinit.m : Initials conditions - userfunc.m : Function

If you choose the option **Ignore and continue**, the **Start** button will disappear. Change directory using **Browse** button or Matlab directory explorer or create those file and press **Read** button in order to reactivate the **Start** button. **Create a new project** and **Go to a Demo Problem** options are described in **Chapter V**.

- **Stop**: Stop the optimization process. The program finishes the last iteration processed an display the following warning message:

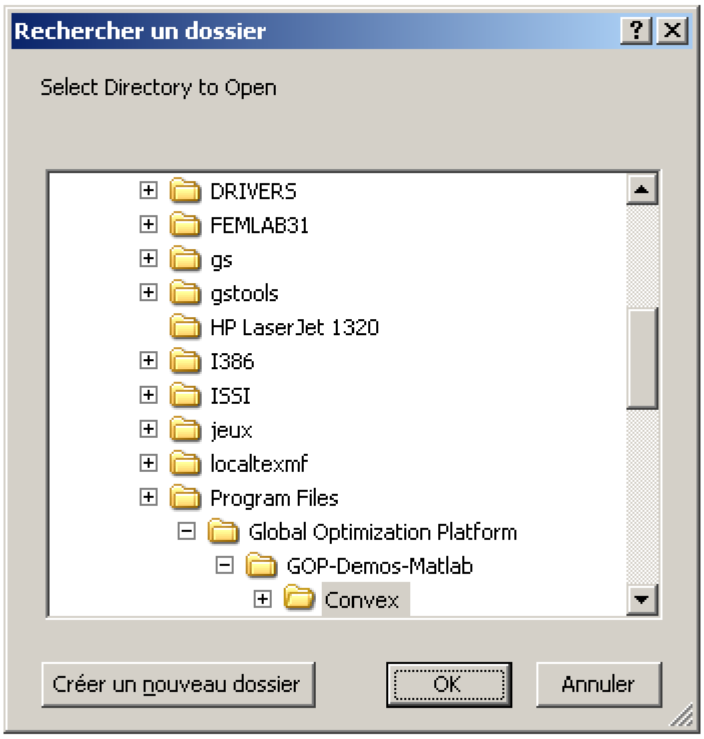

Stop warning

Optimization process stopped by user...

Important note: In order to immediately stop the optimisation process without waiting the end of the iteration you can use the **Crtl+c** or **Matlab break button**. However Matlab may crash. Use this option at your own risk.

Read: Re-read current folder in order to check and reload **userinit.m** and **userfunc.m** files. If one of those file is missing the **Start** button important note is also valid in this case.

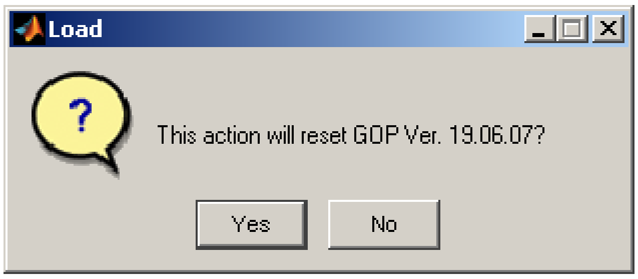
- **Browse**: Change working directory. The following Windows should open:



Choose the directory you want work in and press **OK**. Click **Create a new directory** to create a new directory. Press **Cancel** to stay in the to current directory. When you change the working directory, GOP automatically verify if **userinit.m** and **userfunc.m** are present. If one of those file is missing the **Start** button important note is also valid in this case.

- **Save:** Most optimization options are saved in the working directory. Furthermore the Matlab script *'scriptgop.m'* is created in order to run an optimization with the current option using GOP script commands. See **Chapter III** for more details.

- **Load:** Saved options present in the working directory are loaded. The following question is displayed:



This action will reset GOP Ver. 19.06.07?

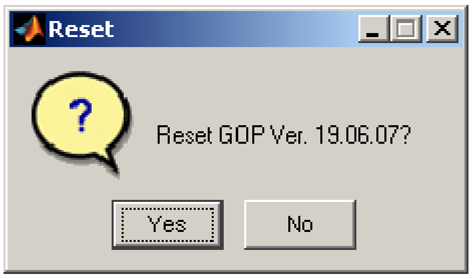
Choose **Yes** to reset and load the options and **No** to cancel.

Important note: Many variable are cleared! Save your results before using this button.

- **Del. Sv.:** Delete the Saved options file.

- **Help:** Display the software documentation..

- **Reset:** Reset the software. The following question is displayed:

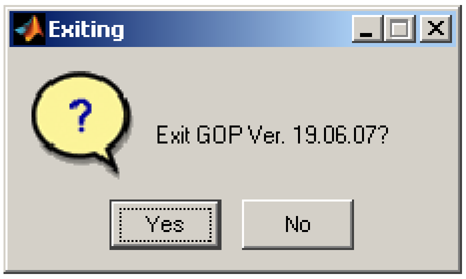


Reset GOP Ver. 19.06.07?

Choose **Yes** to reset the software and **No** to cancel.

Important note: Many variable are cleared! Save your results before using this button.

- **Exit**: Software is closed. The following question is displayed:

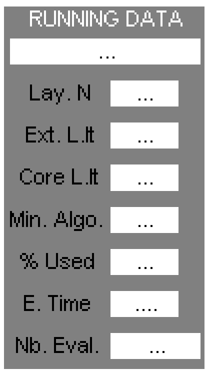


Exit GOP Ver. 19.06.07?

Choose **Yes** to exit the software and **No** to cancel.

c. Zone 2- Running Data

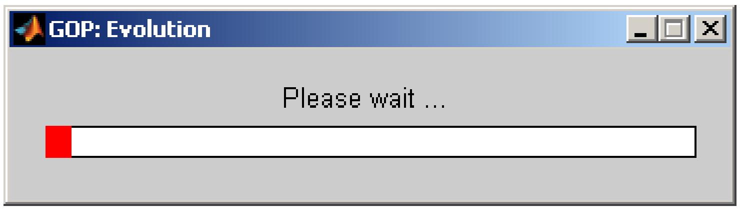
Display the optimisation process iteration and time data:



Options:

- **Lay. N**: External layers executed.
- **Ext. L.It**: Current Layer iteration.
- **Core L.It**: Internal layer iteration.

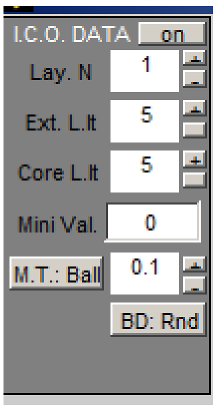
- **Min. Algo.**: Core optimisation Algorithm iteration.
- **% Used**: Total progress of the optimization. A progress bar **GOP: Evolution** is also displayed.



- **E.Time**: Elapsed time since the beginning of the optimization.
- **NB. Eval.:** Number of functional evaluations.

d. Zone 3- Initial Condition Optimization (I.C.O.) data

Corresponds to the external layer algorithms data. Those algorithms intend to improve the initial condition of the core optimization algorithm selected in **Zone 4.**:



Lay. N	1
Ext. L.It	5
Core L.It	5
Mini Val.	0
M.T.: Ball	0.1

Options:

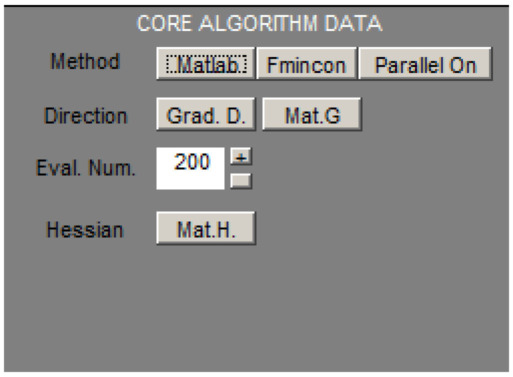
- **On/Off button**: Activate or deactivate the ICO algorithm. If *Off* the other option are hidden and only one run of the core layer algorithm is performed.
- **Lay. N**: Number of external layers.
- **Ext. L.It**: Iteration number of each external layer.
- **Core L.It**: Iteration number of the internal layer.
- **Mini Val.:** Calibration of the ICO algorithm based on a multi-layer secant method. By default the lower boundary of the function to be minimized is set to 0 (default secant method). However, if user has a more precise approximation of this lower boundary (for negative functions, etc.) it should change the value of this option to the adequate lower boundary.
- **M.T.: Ball**: Technique for choosing the second initial point:
  - **M.T.: Ball**: Point is chosen in a ball of center first initial point and with a radius proportional to the search space length using the bottom coefficient (In previous figure 0.1).
  - **M.T.: Rnd**: Point is chosen randomly in the search space.

- **M.T.: Attr**: Only used when a **genetic method** is selected as core optimization method: The second population is created using an secant method starting from each individual and looking in the direction of the best current point

- **BD:** If the ICO algorithm is blocked on a point of the admissible space (for instance, a corner of this space):
  - **BD: Rnd**: Consider a new random initial point.
  - **BD: Stp**: Stop current layer and go to the next layer.
  - **BD: Nt**: Continue from the same point.

e. Zone 4- Core algorithm Data

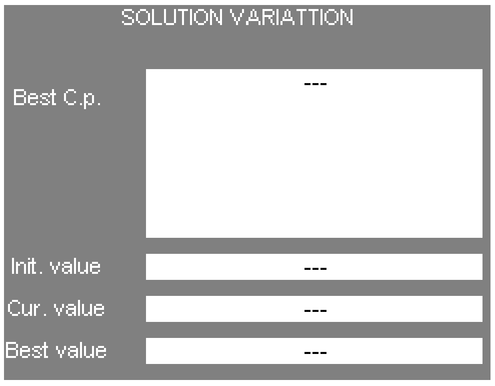
Allows to choose to the core optimization algorithm and specify its options:



The use of this box is detailed in paragraph II.k.

f. Zone 5- Solution Variation

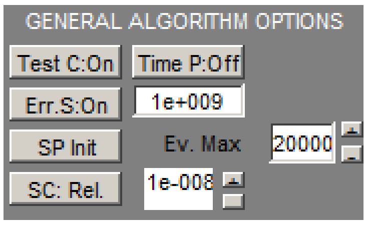
Display functional value evolution during optimisation process:



- **Best C.p.**: Display current best point.
- **Init. Value**: Display initial value of the function.
- **Cur. Value**: Display current value of the function.
- **Best value**: Display current best value of the function.

g. Zone 6- General Algorithm Options

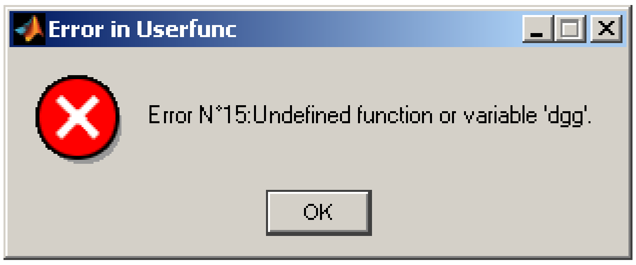
Options of the general optimization process which:



- **Test C.**:
  - If **on**: All the computed points are saved in the workspace in the matrix **savalu** and their value in the vector **valu**. At the end of the optimization they are saved into **Computedpoints.csv** in the save directory **Dir.sav.** defined in **zone 8**. The structure of those variables is described in **Advanced Use** paragraph. Furthermore, if a point is recomputed the software take its previous value. A warning box will display the number of time a computed pointed has been considered.
  - If **Sv**: As previously computed points are saved but the functional value is re-evaluated each time.
  - If **off**: Computed points are not saved.

- **Time P:**
  - If **on**: After each computation a 0.1 sec pause is performed. This allows Matlab to refresh the graphical interface.
  - If **off**: Pause is disabled.

- **Err.S**:
  - If **on** : Software stop in case of functional evaluation error.
  - If **off**: During the functional evaluation if an error occurs the software doesn't stop. The functional value is set to the value defined in the box at the right of **Err.S** button (default value1e9). An error box will display the number and type of errors:


Error in Userfunc

Error N°15:Undefined function or variable 'dgg'.

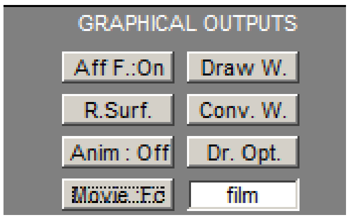
- **SC**.: Stopping criterion
  - **SC Rel**: Expected cost function reduction relatively to its initial value.
  - **SC Abs**: If cost function value is lower than the given value the algorithm stop.
- **SP Init**: Starting point option:

- **SP Init**: Starting point is determined using the **userinit.m** script.
- **SP Rand**: Starting point is created randomly in the admissible space.
- **SP Opt**: Starting point is the previous optimised points. A first optimisation process must have been performed in order to display this option.

- **Ev. Max**: Maximum number of functional evaluation

h. Zone 7- Graphical outputs

Parameters of figure, movie,… displayed during optimisation process:



- **Aff. F.**: This option is disabled if Matlab figure **FIGUSER** is not defined (See **Chapter III**).
  - If **on**: Graph, defined by user, is drawn at each iteration in **FIGUSER**, except during gradient evaluation.
  - If **off**: Graph is not drawn and figure FIGUSER is hidden..

- **Movie**: This option is disabled if Matlab figure **FIGUSER** is hidden.
  - If **on**: A movie is saved (**avi** format) showing graph evolution at each iteration of the external layer algorithms. The movie is saved in the file, named in 'text zone', and directory defined in **Zone 5**.
  - If **off**: Movie is not saved..

- **Conv.W.**: Show/hide the **Convergence Graph window**. This tool is described in Chapter **II.l.**

- **Draw**: Show/hide the functional **Drawing window**. This tool is described in section Chapter **II.m.**

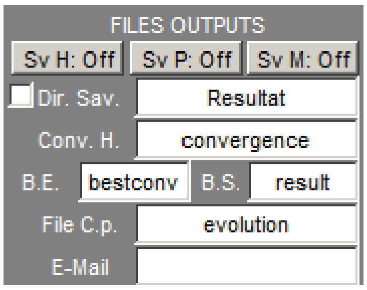
- **R. Surf**: Show/hide the response **Surface drawing window**. This tool is described in Chapter **II.n.**

- **Draw Opt.**: Show/hide drawing option toolbar in the **Drawing window**.

- **Anim**:
  - If **on** : At each functional evaluation each graph is updated.
  - If **off** : Some graphs are not updated until the end of the optimization.

i. Zone 8- Files outputs

Modify options of directory and files where output are saved:



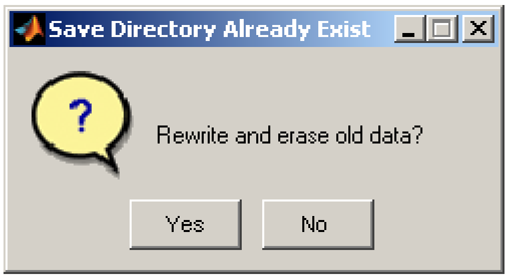
- **Sav H.**: Save convergence history :
  - If **on**: Files are saved after each functional evaluation.
  - If **off**: Files are saved only at the end of the optimisation.

- **Sav H.**: Matlab memory after each functional evaluation:**on** / **off**.

- **Sav C.P.**: Save computed point values:
  - If **All**: All computed points are saved in the file defined by **File C.p.** box.
  - If **Best**: Best points are saved in the file defined by **File C.p.** box.
  - If **off**: No point is saved.

- **Dir.Sav.**: Name of the directory where file are saved Check the **left box** in order to add to this name to the current system date.

  Important note: When you start optimization, if the directory already exist the following dialog box will appears:



  If you choose **Yes** all files in this directory will be erased. No cancel the optimization process.

- **Conv.H.**: Name of the file where the evolution of the functional value is saved.

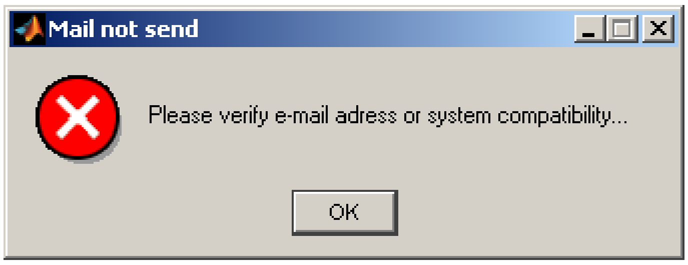
- **B. E.**: Name of the file where the evolution of the best functional value is saved.

- **B. S.**: Name of the files where the best point and its functional value is saved.

- **File C.p.**: Name of the files where computed points and their functional values are saved.

- **E-mail**: E-mail address where the results are sent at the end of the optimization progress.

  Important note: If the e-mail is not send the following error message appears:



Mail not send

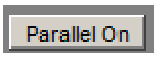
Please verify e-mail adress or system compatibility...

OK

Verify your e-mail address or reconfigure Matlab /operating system default messenger.

j. Core algorithms options

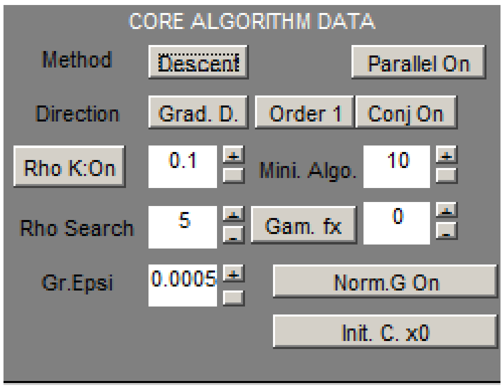
We are interested in this part to describe the different core algorithms and their options implemented in the software. Those options are included in **Zone 2** of the main control panel.



PARALLEL OPTION (on/off): A general option, depending on the selected algorithm, is **PARALLEL** Computing, it is available if the *Matlab® parallel computing toolbox* is installed. It run the parallel version of some algorithms: Steepest descant, fmincon, Genetic Algorithm, Pattern Search,... Please read the Matlab® Help document of this toolbox in order to know the advantages/disadvantages of using parallel computing.

There are four main classes of methods which can be selected pressing the method button.

- **Descent methods:** Those methods are discrete versions of a first or second order dynamical system. They include algorithms such as the steepest descent, heavy ball method, and random direction descent.



CORE ALGORITHM DATA

Method: Descent — Parallel On

Direction: Grad. D. — Order 1 — Conj On

Rho K:On 0.1 — Mini. Algo. 10

Rho Search 5 — Gam. fx 0

Gr.Epsi 0.0005 — Norm.G On

Init. C. x0

Options:

- **Direction**: Choose the descent direction:
  - **Grad.D.**: Use the cost function gradient descent direction.
  - **Rand.D.**: Use a random descent direction.

Important note: At each descent direction evaluation a global Matlab variable **Gradcomp** is set to 1. This global variable is set to 0 when performing other kind of functional evaluation. You can use this variable, for instance, to draw at each intermediate optimization (out of gradient evaluation) step the current solution.

Depending on the selected gradient direction, you can modify the following options:

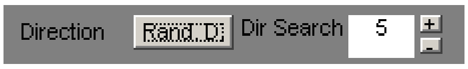
In **Grad.D.** case:
- **Order**: Modify the type of gradient computation:
  - **Order 1**: First order Taylor approximation of the gradient.
  - **Order 2**: Second order Taylor approximation of the gradient.
  - **Matlab** Use Matlab default approximation method (option activated depending of the core optimisation method selected).
  - **User**: User defined gradient (if **usergrad.m** has been created, see **Chapter III.** For more details).

- **Conj**: use a conjugate direction: **on|off**.

- **Norm**: Normalize the gradient direction: **on|off**.

- **Gr.Epsi**: In case of **Order 1** or **Order 2** Taylor approximation of the gradient, choose the epsilon step for finite difference (normalized with the search space length).

In **Rand.D**. case:


Direction Rand. D. Dir Search 5

-**Dir. Search**: allows choosing the maximum number of directions tested in order to have a suitable descent direction.

-**Rho search**: Iteration number of the dichotomy search method used in order to determine an appropriate descent step size ρ.

- **Rho init**: Initial value of the **descent weight coefficient** ρ in the method.

- **Rho K**: Define the method to choose initial value of ρ after the first iteration:
  - If **on**: The initial value of ρ is set to its previous value.
  - If **off**: The initial value of ρ defined in **Rho** init is always used.
  - If **Aut**: The initial value of ρ is fixed automatically.

- **Mini Algo.**: Iteration number of the descent algorithm.

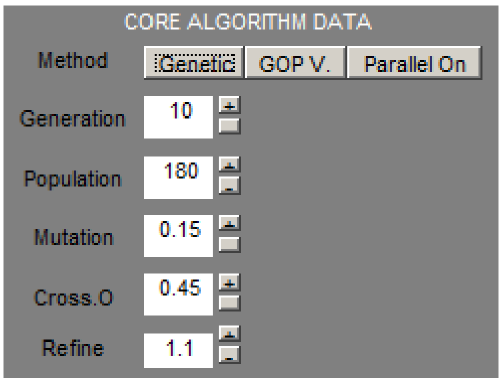
- **Gamma**: Weight coefficient of the second order dynamical system term. If Gamma=0 the considered system is a typically first order system. Else it's a second order dynamical system.

- **Norm G.**: Normalization of the gradient norm:
  - If **on**: The gradient norm is set to 1.
  - If **off**: The gradient norm is not normalized.

- **Init C.**: Varying initial condition:
  - If **x0**: initial point is considered as a new variable to be optimized.
  - If **x'0**: initial Velocity is considered as a new variable to be optimized.

- **Genetic methods**: Use genetic algorithm method.

  - **V.**: Version of the algorithm:
    - **GOP V.**: Use a version specific to GOP.
    - **Matl V.**: Use the version included in **Matlab Global optimisation** toolboxes (this option is available only if this toolbox is activated in Matlab .


CORE ALGORITHM DATA

Method: Genetic | GOP V. | Parallel On

Generation: 10

Population: 180

Mutation: 0.15

Cross.O: 0.45

Refine: 1.1

Options:

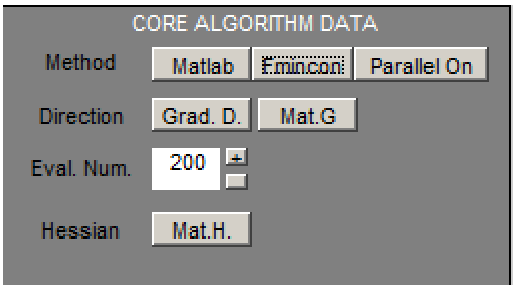
- **Generation**: Generation number.
- **Population**: Population size.
- **Mutation**: Probability of mutation.
- **Cross.O**: Probability of Cross Over.
- **Refine**: Refinement parameter used for the distance of hamming (only for GOP version of the algorithm).

Important note: For previous methods the stopping criterion is the one defined by the option **Stop Crit** present in **Zone 3**.

- **User method**: Use a core optimisation method implemented by user. This option is proposed only if **useropt.m** has been created. **See Chapter III.** for more details.

- **Matlab methods**: In this class we have implemented five optimization methods included in **Matlab Optimization** and **Matlab Global optimisation** toolboxes. You can read Matlab help file to have more explanations of each methods.

  Important note: You license of Matlab should includes those toolboxes in order to use them. In other case those method will not be displayed.



Options:

- **Matlab algorithms:**
  - **Fmincon**.
  - **Lsqnlin**: They are two sub-methods: **Gauss-Newton** or **Levenberg-Marquart**.
  - **Fminunc**: there are three sub-methods: **BFGS, Davidon-Fletcher-Powell** and **Steepest -Descent**.
  - **Fminsch.**
  - **Pattern Search.**
  - **Simulated annealing.**

- **Eval. Num.:** Number of core algorithm iterations.

- **Direction**: Choose the descent direction. Same use and options as the Desent method case.

- **Hessian**: Choose the method to approximate the Hessian of the cost function:
  - If **Math.H.**: Use Matlab default approximation method.
  - If **User**: Use a specific user approximation method 5this option is activated only if **userhess.m** has been created by user, see **Chapter III.**)

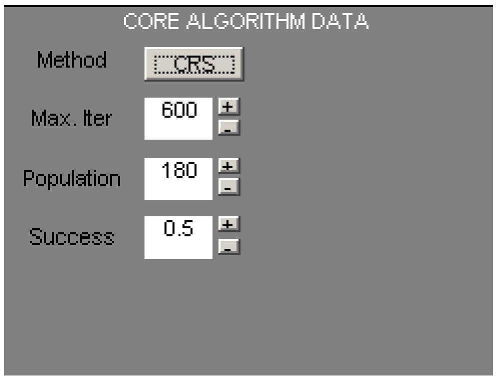
- **Additional Matlab Options and constraints:** Depending on the Matlab algorithm selected (see they respective Help Guide), you can use the following variables, for example defined in *userinit.m* (See **Chapter V**), to define additional options or restrictions compatible with those algorithms:

  - *umra (Matrix) and umrb (Vector)*= Matlab constraint umra*x<=umrb

  - *umraeq (Matrix) and umrbeq (Vector)*= Matlab constraint umraeq*x=umrbeq

  - *umrnlc*= Matlab vector function for non linear constraint umrnlc(x,1)<=0 and umrnlc(x,1)=0

  - *usermatopt*= additional Matlab optimization algorithm options (see Matlab User Guide)



- **Controlled Random Search methods**: Use genetic algorithm method.

  - **Max. Iter**: Maximum number of functional evaluation
  - **Npop:** Population size
  - **Success:** Success rate.

k. Graphical toolboxes

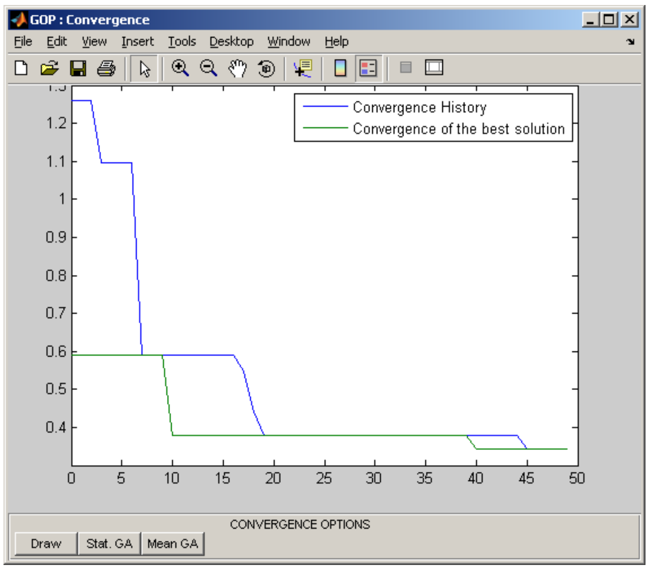
We will now see how to use the various graphical tools implemented in GOP. All those Windows contains the classical Matlab menu in order to modify and save them. The option **Dr. Opt.** in **Zone 3** is active for those figures and allows to Show/Hide the figure menu.

Convergence Windows

Click on the **Conv W.** present in **Zone 3**. A new window entitled **GOP: Convergence** should appear. The option **Anim** in **Zone 3** is active for this window.

This Windows display two graphs:

- The **Convergence History**: Blue line corresponds to the convergence history of the cost function during optimization.
- The **Convergence of the best element**: Green line corresponds to the evolution of the cost function value of the best element found during optimization.



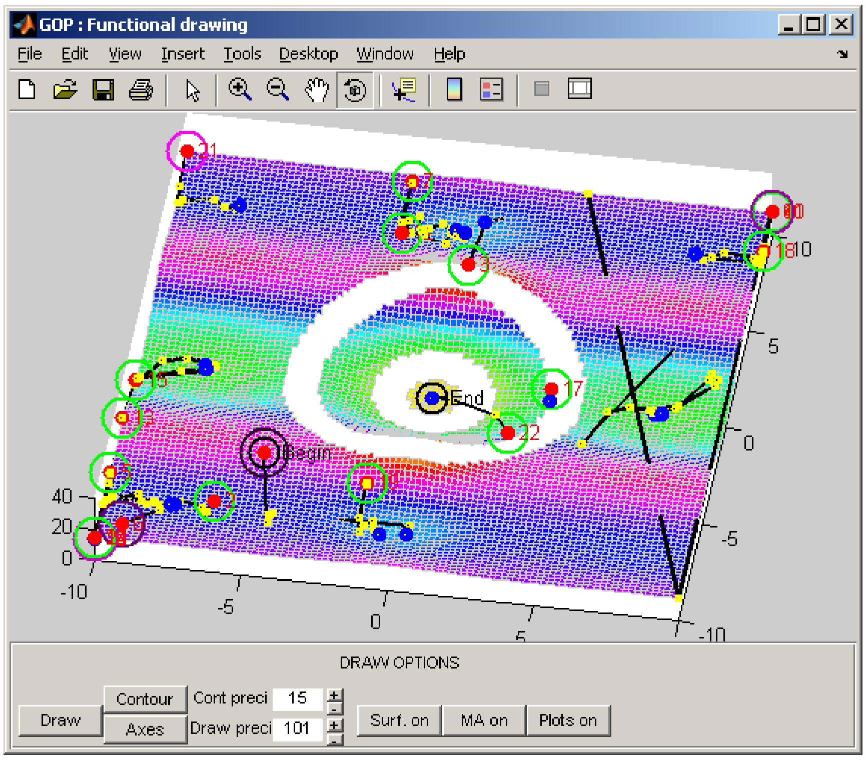
Options:

- **Draw**: Draw the convergence histories of the last optimisation process.
- **Stat. GA**: Draw the surface of the cost function value of each individual in the GA population at each iteration (only for GAs).
- **Mean GA**: Draw the evolution of the mean of individuals cost function value (only for GAs).

- **Functional drawing**

Click on the **Draw W.** present in **Zone 3**. A new window entitled **GOP: Functional drawing** must appear. The option **Anim** in zone 3 is active for this window.

This figure allows to display the considered cost function in the case when the dimension of the problem is inferior or equals to 3 and propose a dynamical graphical representation of the optimisation process.



Cont preci 15
Draw preci 101

Options:

- **Draw**: Drawing the cost function.
- **Axes**: Show/Hide figure axes.
- **Contour**: Show/Hide iso-contours of the function (only in dimension 2).

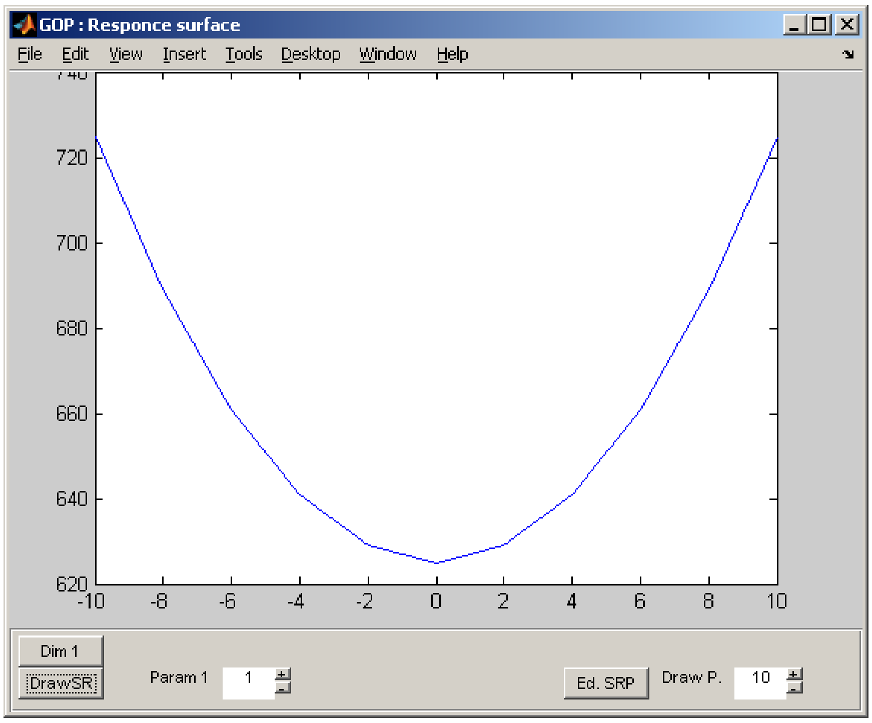
- **Draw preci**: Selected the mesh precision for the cost function graph.
- **Cont preci**: Number of iso-contours lines. Other options are applicable in order to show the graphical evolution of the optimization algorithm (only in dimension 2).
- **Surf**: Show/Hide the functional graph.
- **MA**: Show/Hide the core algorithm trajectory evolution with a black line.
- **Plots**: Show/Hide the plots showing the principal points computed by the core algorithm:
  - Red point: Initial condition of core algorithm.
  - Yellow point: Intermediate point of core algorithm.
  - Blue point: Final point of core algorithm.
  - Circle: Initial condition of external layer algorithm.

  For each previous element a number indicate its order of apparition.

- **Response surface drawing**

Click on the **R. Surf.** present in **Zone 3**. A window entitled **GOP: Response surface** appears.

This figure allows to display a surface response of the considered cost function.



Depending on the dimension of the problem, options are displayed:

- **Dim**: Choose the dimension of the drawing
- **Draw SR**: Draw the surface of response of the cost function.

- **Param 1**: Choose the first dimension to be studied.
- **Param 2**: Choose the second dimension to be studied.
- **Param 3**: Choose the third dimension to be studied.
- **Ed. SRP**: Edit the vector of fixed dimension value.
- **Draw P.**: Selected the mesh precision for the cost function graph.

III. Script version guide

This version of GOP is designed in order to use it in text I/O systems. In fact, this is an conversion of the GUI interface in command lines. The lecture of previous chapter is strongly recommended in order to better understand the use of the script version. As you will see in this chapter, GOP options can be configured using particulars variables and command lines.

a. GOP Script main commands

The principal command in order to control GOP Script are:

- gopscr: Initialize GOP Script. **This command must be executed before the use of other GOP Script commands.**
- goprun: launch the optimization process.
- gopshelp: display GOP script command list.
- gopvarlist: display GOP Script variables name list and their current values.
- gopoptsum: display current GOP Script options and parameters.
- gopoptsave: save GOP Script options in the current directory in '**gopoptions.mat**'.
- gopoptsavscr: save a Matlab script with the current GOP option in the current directory in **'scriptgop.m'** in order to run a GOP script optimization process.
- gopoptload: load GOP Script options file presents in the current directory.
- gopoptdel: delete GOP Script options file presents in the current directory.

b. GOP Script variables

Those variables allows ton configure the options of the optimization process. A list of those variables can be displayed using the command 'gopvarlist'. All of those variables take a default value (see 'gopvarlist') when 'gopscr' is initialized.

There are seven set of variables. Each one corresponding to options included in one of the GOP GUI zone (See previous chapter):

- General Algorithm Options

  spinit=Initial condition 0=vinit 1=rnd 2=xoptg
  epsijs= stop criterium | Current Value=1e-008
  stopcrit= type of stop criterium 0=proportional reduction 1=value to reach 2=benchmark test
  reachpoint= Point to reach | e1reachpoint=precision 1 | e2reachpointt=precision 2 (for benchmark test case)
  evalumax= Number of maximum funcitonal evaluation

- Core Optimization Method Options (Core Algorithm Data GUI zone):

  mopt=Optimization Method 1=Descent 3=MATLAB 4=GA 5=User
  goppar=Activate the parallel computation of some components of GOP (Gradient, GA,...) 0=Off 1=On

- GOP descent method options (Core Algorithm Data GUI zone):

  nbgrad= iteration number
  method= descent direction: 1=gradient 2=random
  retdm= number of random direction retry
  normgrad= normalization of the gradient 1=on 0=off
  conj= conjugate gradient 1=on 0=off
  gam200= ordre of SDM
  iordre= gradient type 1= Taylor 1 point 2 =Taylor 2 points 3=User 4=Matlab
  epsfd= Taylor gradient perturbation
  rho00=rho initial
  autorho=Automatic selection of the rho coefficient: 1=on 0=off
  srho= number of iteration for the search of rho
  initcond2=initial condition in case of a 2nd order descent system: 0=x0 1=x'0

- MATLAB methods options (Core Algorithm Data GUI zone):

  mtool= Matlab algorithm used 1=fminsearch 2=fmincon 3=lsqnonlin 4=fminunc 5=Patternsearch 6=CRS
  mateval= Iteration Number
  iordre= Gradient type 1= Taylor 1 point 2 =Taylor 2 points 3=User 4=Matlab
  epsfd= Taylor gradient perturbation
  hess=Hessian Version for fmincon 1=User 0=Matlab
  mopt2=Hessian Method for fminnunc 0=BFGS 1=SD 2=DFP
  mopt3=Hessian Method for lsqnonlin 0=Levenberg Marqurt on /off
  umra (Matrix) and umrb (Vector)= Matlab constraint umra*x<=umrb (should be defined in unserinit.m)
  umraeq (Matrix) and umrbeq (Vector)= Matlab constraint umraeq*x=umrbeq (should be defined in unserinit.m)
  umrnlc= Matlab vector function for non linear constraint umrnlc(x,1)<=0 and umrnlc(x,1)=0 (should be defined in unserinit.m)
  usermatopt= additional Matlab optimization algorithm options (should be defined in unserinit.m)

- GA method options (Core Algorithm Data GUI zone):

  gaver= Version of GA 0=GOP 1=Matlab
  Npop= Population Size
  Ngen= Generation Number
  pm= Mutation Probability
  pc= Cross Over Probability
  Refb= Refinement Parameter

- CRS method options (Core Algorithm Data GUI zone):
  Npop= Populaiton Size
  CRSMAX= Maximum funcitonal evaluation
  pm= Success Rate

- Multi-Layer Algorithm Options (Layer Data GUI zone):

spinit=Initial point 0=defined in userinit 1=random 2=previous optimized point
nbext2= number of layers
nbext1= iteration number of the external layers
nbbvp= iteration number of the internal layer
methodt= Method for choosing second point : 1=ball 2=rnd 3=attr
radb= radius of the search ball
avrep= In case of blocked linear search: 1= Generate another random direction from last point, 2= Stop current line search algorithm, 0= Continue

- Output Options (Files and Output options GUI zone):

savedir= Output directory Name
fsdate= if =1 Date is added to Output Directory Name
fileresult=Result File Name
filevconv=General Convergence File Name
filebconv=Best Element Convergence File Name
savefile= convergence save frequency 0 Real time save 1 save after each main iteration
elevo=Computed Points File Name
login=Computed Points are 0=not saved 1= All points are saved 2= Only best points are saved
filefilm=Movie File Name
mailto=User E-Mail adress where results are sent

- Processing Options (Processing options GUI zone):

savemem= Save matlab memmory? 1=yes 0=No
teste= Are recomputed point are re-evaluated? 1=yes 2=Only Saved 0=No
pausette= Make a pause between each iteration if =1
passerr= Stop in error computation if =1
passerrval= Functional Value in cases of errors.

Those variables can be saved in the current directory using the ‘gopoptsave’ command. Then you can download them using the ‘gopoptload’ command (in the same directory). The option file can be deleted using the ‘gopoptdel’ command.

Once you have configured the optimization process you can display this configuration using the ‘gopoptsum’ command.

Example:

Executing ‘gopscr’ and entering the following variables:

```
mopt=1;
nbgrad=50;
method=1;
normgrad=1;
conj=1;
gam200=0;
iordre=1;
```

```
epsfd=1e-6;
autorho=1;
srho=30;
spinit=0;
nbext1=1;
nbext2=5;
nbbvp=5;
mthodt=2;
teste=0;
```

‘gopoptsum’ return the following output:

```
***********************
* GOP Options summary *
***********************

Multi-Layer Algorithm Options
*****************************

Number Of External Layer: 5
Iteration Of External Layer: 1
Iteration Of Internal Layer: 5
Stopping Criterium: 1e-008
First Initial Condition: Fixed in userinit
Optimization method: ERROR

Core Optimization Method Options
***********************************

Method: GOP Descent Method
Iteration Number: 50
Second Order Coefficient: 0
Optimized Initial Condition: x0
Descent Direction: Gradient
Gradient Type: Taylor 1
Perturbation Parameter: 1e-006
Conjugate: On
Direction norm: Normalized
Initial Rho: Automatic
Rho Search Iterations: 30

Output Options
*****************************

Output Directory: Resultat
Result File: result
General Convergence File: convergence
Best Element Convergence File: bestconv
Convergence Write Frequency: Real Time
Computed Points File: evolution
Points Saved: None
Movie File: film
User E-Mail Adress: None

Processing Options
*****************************
```

```
Verifying Computed Points: No
Pause Between Each Cost Funciton Evaluation: No
Stop Proces If Error Occurs During Funcitonal Evaluation: No
Function Value In Cases Of Errors: 1000000000
```

c. Create an automated optimization script

In order to create an automated Matlab script (for example to use it in a queued process):

1- A simple way to create an automated script is to use the SAVE option of the GOP GUI interface or use the gopoptsavscr command to create a script called *'scriptgop.m'* which contains the current GOP options.

2- You can also create this script manually. To do so, you should respect the following m-file structure:

```
%%% GOP Script initialization
gopscr
%%% GOP Script variables modification (optional)
mopt=…
%%% Verification of the options (optional)
gopoptsum
%%% Running optimization process
goprun
%%% Exiting Matlab options (optional)
exit
```

IV. Program and Variables Architecture

In this chapter we describe the structure of the Matlab version of GOP, the files required to create a personal optimisation problem, the main GOP Matlab variables and the output files. The objective of this chapter is to allow user to have an advanced understanding of the functioning of GOP.

a. GOP Matlab program structure

Those files present in **GOP-Sources** directory represent the source code of GOP:

- **Launch the software**: gop.m
- **Software initialization**: init.m, bmor.m, Liver.m.
- **Graphical User Interface**:
  - guim.m: Build general Graphical User Interface.
  - guidr.m: Build Drawing window.
  - guicv.m: Build Convergence window.
  - guisr.m: Build surfqce response window.
  - draw.m: Build Functional Draw control panel.
  - drawsr.m: Build Response Surface control panel.
  - action.m: Define button's effect.
  - exiting.m: Define exiting sub-script.
  - pausing.m: Launch pause.
  - saveo.m: Launch option-save sub-script.
  - Loado.m: Launch option-load sub-script.
- **Multi-layer optimization code**: bmot.m, couche0.m, .couche1.m, couche2.m, affresult.m, timec.m, pour.m.
- **Descent algorithm**: gradopt.m,gradient.m, rhodi- cho.m, rdmd.m, miniopt.m.
- **Genetic algorithm**: initag.m, agen.m, selection.m, funcbraggech.m, randselect.m.

b. User files

Those files should be created by user in order to generate its own optimisation problem and solve it with GOP:

- **Userinit.m**: Define initial conditions of the problem.
- **Userfunc.m**: Function to be optimized.
- **Usergrad.m** (optional): User gradient method to be used instead of implemented gradient methods.
- **Userprojdom.m** (optional): Function that project evaluated points in the admissible control space. It's particularly useful when the control space doesn't have a box.
- **Useropt.m** (optional): User core optimization method.
- **Userinterface.m** (optional): Additional graphical interface launched at the beginning.
- **Useraffich.m** (optional): Output to be displayed at the end of the optimization algorithm.
- **Userexiting.m** (optional): Script executed when exiting the software.

A more detailed description of each file is presented in **Chapter V**.

c. Matlab variables

When the program is running various variables are created into the Matlab Workspace. Here we describe the principal ones:

- **ndim**: The dimension of the problem. It's an Integer number.
- **vinit**: The given initial condition. If it's not given, only Random Initial condition option described in section 1.2.1 will be activated. It's a ndim-dimension vector of the form:

Initial value of parameter 1
…
Initial value of parameter ndim

- **xmax**: The upper boundary of the problem. It's a ndim-dimension vector of the form:

maximum value of parameter 1
…
a maximum value of parameter ndim

- **xmin**: The lower boundary of the problem. It's a ndim-dimension vector of the form:

minimum value of parameter 1
…
minimum value of parameter ndim

- **xoptg**: Best solution found during optimization process. It's a ndim-dimension vector of the form:

Optimal value of parameter 1
…
Optimal value of parameter ndim

- **costsaveming**: Functional value of xoptg.
- **evalu**: Number of functional evaluation. It's an Integer number.
- **timegui**: Computational time during optimization. It's a Real number
- **savalu**: Elements computed during optimization process. It's a ndim×evalu-matrix of the form:

First value of parameter 1	. . .	First value of parameter ndim
. . .	. . .	. . .
evaluT h value of parameter 1		evaluT h value of parameter ndim

- **valu**: Functional value of all computed points during optimization process. It's a evalu-dimension vector of the form:

Value of first point.
. . .

Value of evaluT h point

- **conv**: Functional value history of the local minimum found at each iteration of core-optimization method. It's a iterc-dimension vector of the form:

Value of first local minimum
. . .
Value of itercT h local minimum

where iterc to core-optimization total iteration number.

- **bconv**: Functional value evolution of the best element found during optimization. It's a iterc-dimension vector of the form:

Value of first best element
. . .
Value of itercT h best element

- **MOV**: If film option is enabled, this variable stock user graph evolution picture at the end of each core-optimization method. It has a Matlab movie structure.
- **FIGUSER**: Matlab figure name reserved for user graphs. This figure should be defined in **Userinterface.m**. Note that you can add other Matlab figures for your personal graphs (See **Chapter IV.** for more details).

d. Output files description

During optimisation process, the software stock into the selected folder (See **Chapter II**) various output files:

- **Diary.txt**: Matlab Command Windows outputs are stocked during optimization process, including error messages. This is a log file using a classical **txt** format.
- **datbmo.txt** (obsolete): This file contains all the information about GOP options. It's stocked before running optimization. It's a **txt** format file.
- **result.txt**: Stores at the end of optimization, the best element found and it's functional value. It's a **txt** format file of the form:

Final point:
Value of parameter 1
.
. Value of parameter ndim Functional
value : Functional value

- **result.csv**: We store at the end of optimization, the best element found and it's functional value. It's a (ndim+1)- **csv** format file of the form:

[Value of parameter 1 ,..., Value of parameter ndim, Functional value]

- **convergence.pts**: In this file we store convergence history of the function. It's a **pts** format file of the form:

1	. . .	functional value 1
. . .	. . .	. . .
iter1*iter2*iterc		functional value iter1*iter2*iterc

- **bestconv.pts**: In this file we store history of the best functional value. It's a **pts** format file of the form:

1	· · ·	best functional value 1
· · ·	· · ·	· · ·
iter1*iter2*iterc		best functional value iter1*iter2*iterc

- **computedpoints.csv**: This file contains all computed points and their functional value. It's a (ndim+2)×evalu-**csv** format file of the form:

1	Parameter 1	· · ·	Parameter ndim	functional value 1
· · ·	· · ·	· · ·	· · ·	· · ·
evalu	Parameter 1	· · ·	Parameter ndim	functional value evalu

- **evolution.txt**: We store all the minimum found during core optimization process and their functional value. It's a **txt** format file of the form:
-

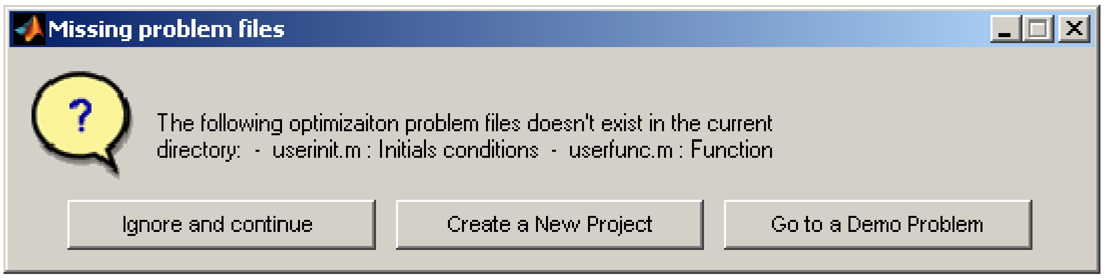
Functional value 1: Functional value of local minimum 1
point 1: Parameter values of local minimum 1
· · ·
Functional value iterc: Functional value of local minimum iterc
point iterc: Parameter values of local minimum iterc

- **Workspace.mat**: Save the workspace with all variables. It's a Matlab **mat**-format file.

V. Building a personal optimization problem

In order to use GOP with your own optimization problem you need to generate various files listed in **Chapter IV.b**.

If those **optimisation problems files** are not present in the working directory, the following message should appear:

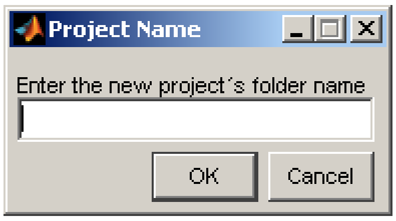


The following optimizaiton problem files doesn't exist in the current directory: - userinit.m : Initials conditions - userfunc.m : Function

If you choose the option:

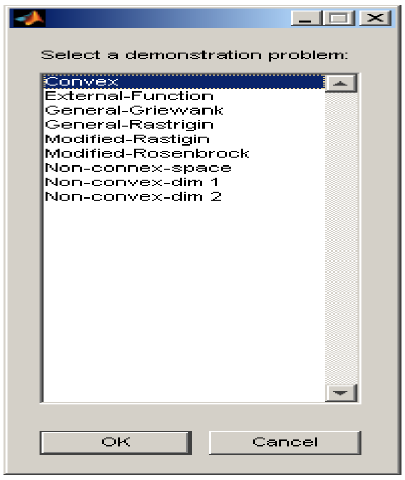
- **Ignore and continue**: GOP will be initialized and you should use the button **Browse** to manually change to a directory where those files are present.

- **Create a new project**: The following prompt will appear:



Enter the name of the project directory where the basic optimization files (**Userinit.m** and **Userfunc.m**) will be created. Those files are incomplete. See next subsection to see how to complete them. In case of problem, an error message appears and you should verify the name of the directory or re-install GOP.

- **Go to a Demo Problem**: The following prompt will appear:



Then select a demonstration optimization problem included in **GOP-Demos-Matlab**. Some of those demos are described in next subsection. In case of problem, an error message appears and you re-install GOP.

a. Building files using a Matlab version of GOP

Important note: We recommend to work in an folder different than **the GOP installation directories** in order to avoid to have your files in a shared directory.

Once you have created this folder, you must build the two following Matlab files:

- **Userinit.m**: This file must have a M-script structure. It should contain the following variables: ndim, xmin, xmax, vinit. Other variables or initialization scripts could be added to this file.

  Example: We can consider a code corresponding to the minimization of the 2- dimensional Rastringin function into the subset [−5, 5]2 starting from [4, 4]:

  ```
  ndim=2;
  for i=1:ndim
  xmin(i)=-5;
  xmax(i)=5;
  vinit(i)=4; end
  ```

  Note: *userinit.m* should be the correct file to also define some restriction compatible with the Matlab algorithms (when they are used as core algorithm) or additional MATLAB algorithm options: see variables umra, umrb , umraeq, umrbeq, umrnlc and usermatopt in **Chapter II.j**:

  - *umra (Matrix) and umrb (Vector)*= Matlab constraint umra*x<=umrb
  - *umraeq (Matrix) and umrbeq (Vector)*= Matlab constraint umraeq*x=umrbeq
  - *umrnlc*= Matlab vector function for non linear constraint umrnlc(x,1)<=0 and umrnlc(x,1)=0
  - *usermatopt*= additional Matlab optimization algorithm options (see Matlab User Guide)

- **Userfunc.m**: This file must have a function-script structure. It should return the value of the cost function. There are three main kind of functions.

  1- A function using Matlab routines.

  Example: Considering the previous Rastringin optimisation problem (in the directory **GOP-demos\Rastringin**):

```
function J=userfunc(x)
temp=0.;
for ii=1:length(x)
temp=temp+x(ii).2-cos(18*x(ii));
end
J=temp+length(x);
end
```

2- A function using an external code (C++, Fortran) routines. You have two solutions:

2.1- You compile your code in an external program that reads an input file written by a Matlab function and writes an output file read by the same Matlab function.

Example: This example is associated to the **External function example** (in the directory **GOP-demos\External**): The compiled program, called **external.exe**, reads its inputs in **Input.dat** and writes its outputs into **Output.dat**. Thus the **userfunc.m** file is of the form:

```
function J=userfunc(x)
fid=fopen('Input.dat','w');
fprintf(fid,'%0.12f \n ',x);
fclose(fid);
!external.exe
fid=fopen('Output.dat','r');
J=fscanf(fid,'%g')+1;
fclose(fid);
!del *.dat
end
```

2.2- Another way consist to compile directly your code using Matlab Mex library (see Matlab help file for more detail) in order to call it directly using a Matlab function.

Example: In this case, an example involving optical fibber synthesis, can be directly asked by e-mail to the authors.

Important note: This method is generally less time-consuming.

2.3- A function using an external black-box (Femlab, Fluent) that can be interfaced with Matlab You can use this interfacing propriety in order to generate an **userfunc.m**.

Example: Femlab can directly generate a m-file from its models. The geometry and equations are generated into this file and then the Femlab solver is called to give a numerical solution. You can incorporate directly this generated file into your "userfunc.m. In this case, an example involving microfluidic mixer shape, can be directly asked by e-mail to the authors.

In addition to **Userinit.m** and **Userfunc.m** user can generate optional files:

- **Usergrad.m**: This file must have a m-script structure. It should build the gradient direction using a ndim-vector variable fpx. Once you have created this file you can selected User gradient option (See previous sec- tion).

  <u>Example:</u> For Rastringin optimization problem (in the directory **GOP-demos\Rastringin**):

```
for ii=1:ndim
fpx(ii)=2*x(ii)+18*sin(18*x(ii));
end
```

- **Userprojdom.m**: This file must have a function-script structure. It take a point in the **defined** control space as input and return the projection of this point on the problem **admissible** control space as first output. It must also return 1 as second output in cases when the projection is successful. When the projection is not possible this function should return 0 as second output.

  <u>Example:</u> An example of a such file is present in the directory **GOP-demos\Non-connex-space**. The admissible space is non-connex, separated by two rings. If the given point is not inside this admissible space, the point is projected using weight coefficients:

```
function [x,bonbord2]=projdom(x)
global fpx x0 xmin xmax evalu PCF2 bound nbound ndim
xx=x;
bonbord2=1;
b1=1;b2=2;
if (norm(x,2)>=b1)&(norm(x,2)<=b2)
 bonbord2=0;
  if (abs(norm(x,2)-b1)<abs(norm(x,2)-b2))
  x=x/norm(x,2);
  x=x*b1;
  else
    x=x/norm(x,2);
    x=x*b2;
  end
end
b1=4;b2=5;
if (norm(x,2)>=b1)&(norm(x,2)<=b2)
 bonbord2=0;
  if (abs(norm(x,2)-b1)<abs(norm(x,2)-b2))
  x=x/norm(x,2);
  x=x*b1;
  else
    x=x/norm(x,2);
    x=x*b2;
  end
end
end
```

- **Useraffich.m**: This file must have a m-script structure. This script is launched ate the end of optimization, in order to perform a post-processing treatment of the optimized solution.

- **Userinterface.m**: This is a m-script file that build additional interface Windows to be interfaced with GOP.

  Important note: The figure name **FIGUSER** is already reserved in Matlab workspace to build your own figure. But you can add other figures at glance

- **Userexiting.m**: Script executed when exiting the GOP.
- **Useropt.m**: In order to solve an optimization problem, maybe you want to integrate your own optimization routine as core-optimization method, more adapted to the considered problem. To do so, you need to create an m-script **Useropt.m** and activate **User method** in **Zone 2 of main control panel**.

  The main structure of this m-script should read:

  – Consider variable x0, a ndim-vector as initial condition of your optimization algorithm.

  – Perform your algorithm

  – Return found solution into x0 and its associated functional value into the real variable f.

  Example: You can find an application regarding the file **gradopt.m** (in the directory **GOP-sources**) that corresponds to the descent method included in GOP.

b. Building files using the Stand-Alone version of GOP

In this case, the user should build the followings files:

- **Init.Dat**: This text file contains the dimension, the boundaries of each dimension and initial points of the problem. It should follows the following structure (comments included)

```
%dimension of the problem
here an integer number
%Boundaries of each dimension (max(1),min(1),max(2),min(2),....)
boundary maximum of first dimension (real number)
boundary minimum of first dimension (real number)
boundary maximum of second dimension (real number)
.
.
%Initial point(Option)
initial point value in the first dimension (real number)
initial point value in the second dimension (real number)
.
.
```

- **Userfunc.exe**: Represent the cost function to be minimized. This file reads its inputs in **Input.dat** and returns the value of the value of the cost function into the output file **Output.dat**.

    - **Input.dat** must have the following structure:

    ```
    Value of the point in the first dimension (real number)
    Value of the point in the second dimension (Real number)
    .
    .
    ```

    - **Output.dat** must have the following structure:

    ```
    Value of the cost function (real number)
    ```

VI. Some References

Full description of the method:

1. Benjamin Ivorra. “Semi-deterministic global optimisation methods and industrial applications”. PhD thesis. University of Montpellier II. (2006) : http://www.mat.ucm.es/~ivorra/these/These-Ivorra.pdf

2. Benjamin Ivorra, Bijan Mohammadi, Patrick Redont and Angel Manuel Ramos del Olmo. “Optimizing initial guesses to improve global search”. Journal of Global Optimization. Submitted. Springer Science.

Short presentation of the method and industrial applications:

3. Benjamin Ivorra, Angel Manuel Ramos del Olmo and Bijan Mohammadi. “Optimization strategies in credit portfolio management”. Journal of Global Optimization. Vol. 43(2): 415- 427. Springer Science (2009) **(area: finance)**

4. Benjamin Ivorra, Angel Manuel Ramos del Olmo and Bijan Mohammadi. “A Semi-Deterministic Global Optimization Method. Application to a Control Problem of the Burger Equation, comparing with Other Methods”. Journal of Optimization Theory and Applications. Vol. 135(3):549-561. Kluwer Academic (2007) **(area: control problem)**

5. Benjamin Ivorra, David E. Hertzog, Bijan Mohammadi, Juan G. Santiago. “Semi-deterministic and genetic algorithms for global optimization of microfluidic protein-folding devices “. International Journal for Numerical Methods in Engineering. Vol. 66(2): 319- 333. Wiley InterScience (2006) **(area: fluid dynamic)**

6. Benjamin Ivorra, Bijan Mohammadi, Patrick Redont, Laurent Dumas, Olivier Durand. “Semi-Deterministic vs. Genetic Algorithms for Global Optimization of Multichannel Optical Filters”. International Journal of Computational Science and Engineering. Vol. 2(3): 170- 178. Inderscience publishers (2006) **(area: optical fiber)**

7. Larvi Debiane, Benjamin Ivorra, Bijan Mohammadi, Frank Nicoud, Thierry Poinsot, Alexandre Ern, Hernst Pitsch. “A low-complexity global optimization algorithm for temperature and pollution control of a complex chemistry flame”. International Journal of Computational Fluid Dynamics.Vol. 20(2): 93-98. Taylor & Francis (2006) **(area: combustion)**

8. David E. Hertzog, Benjamin Ivorra, Bijan Mohammadi, Olgica Bakajin, and Juan G. Santiago. **“**Optimization of a Fast Microfluidic Mixer for Studying Protein Folding Kinetics”**.** Analytical chemistry. Vol. 78(13): 4299-4306. ACS Publications (2006) **(area: fluid dynamic - experimental validation of paper 6)**

9. Benjamin Ivorra, Bijan Mohammadi, Laurent Dumas,Olivier Durand. “Semi-Deterministic Recursive Optimization Methods for Multichannel Optical Filters”. Numerical Mathematics and Advanced applications. 957-964. Springer Science (2006) **(area: optical fibber)**

VII. Credits

Mathematical group research:

Ivorra Benjamin (Universidad Complutense de Madrid)
Bijan Mohammadi (CERFACS, Toulouse)
Angel Manuel Ramos del Olmo (Universidad Complutense de Madrid)
Patrick Redont (Université de Montpellier 2)
Jean-Paul Dufour (Université de Montpellier 2)

Developers:

User interface:
Ivorra Benjamin

Semi-determinist optimization algorithm:
Bijan Mohammadi
Ivorra Benjamin

Genetic Algorithm:
Laurent Dumas (Université de Paris 6)
Ivorra Benjamin

Controled Random Search:
Juana Lopez Redondo (Universidad de Almería)
Pilar Martínez Ortigosa (Universidad de Almería)
Ivorra Benjamin

Debuggers:

Ivorra Benjamin
Bijan Mohammadi
Angel Manuel Ramos del Olmo
Damien Isèbe (Université de Montpellier 2)
Juan-Antonio Infante (Universidad Complutense de Madrid)
Jose Maria Rey Cabezas (Universidad Complutense de Madrid)

VIII. Contact information

Pr. Ivorra Benjamin

Adress : Departamento de Matemática Aplicada
Facultad de Ciencias Matemáticas
Universidad Complutense de Madrid
Avda. Complutense s/n
28040 Madrid. España
Phone : +34 91.394.44.15
Fax : +34 91.394.46.13
e-mail : ivorra@mat.ucm.es
Webpage : www.mat.ucm.es/~ivorra

Pr. Bijan Mohammadi

Adress : CERFACS, Toulouse, France
e-mail : bijan.mohammadi@cerfacs.fr

Pr. Angel Manuel Ramos del Olmo

Adress : Departamento de Matemática Aplicada
Facultad de Ciencias Matemáticas
Universidad Complutense de Madrid
Avda. Complutense s/n
28040 Madrid. España
Phone : +34 91.394.44.80
Fax : +34 91.394.46.13
e-mail : angel@mat.ucm.es
Webpage : www.mat.ucm.es/~aramosol