

# Gnuplot OX server Manual

---

Edition : auto generated by oxgentexi on 15 May 2012

OpenXM.org

---

# 1 GNUPLOT Functions

This chapter describes interface functions for GNUPLOT ox server `ox_sm1_gnuplot`. These interface functions are defined in the file `gnuplot`. The file ‘`gnuplot.rr`’ is at ‘`$(OpenXM_HOME)/lib/asir-contrib`’.

```
[255] gnuplot.start();
0
[257] gnuplot.gnuplot("plot sin(x**2);");
0
```

The function `gnuplot.heat(dt,step)` demonstrates our gnuplot interface. It numerically solves the heat equation

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2}, \quad u(t, 0) = u(t, 1) = 1$$

with the initial condition

$$u(0, x) = x, \quad (0 \leq x \leq 0.5), \quad u(1, x) = 1 - x, \quad (0.5 \leq x \leq 1)$$

by the explicit scheme for  $0 \leq t \leq dt * step$ . The segment  $[0,1]$  is divided into `Heat_N` segments. The static variable `Heat_N` can be set by the function `gnuplot.heat_set_N`. If the celebrated Courant-Friedrichs-Levi number  $dt * Heat\_N * Heat\_N$  is less than or equal to 0.5, then the explicit scheme is numerically stable. One can observe the instability by changing CFL number.

```
gnuplot.heat_set_N(20); gnuplot.heat(0.001,30);    (CFL number is 0.4)
gnuplot.heat_set_N(20); gnuplot.heat(0.003,30);    (CFL > 0.5 unstable)
```

Author of GNUPLOT: Thomas Williams, Colin Kelley. <http://www.gnuplot.info>

## 1.1 Functions

### 1.1.1 gnuplot.start

```
gnuplot.start()
:: Start ox_sm1_gnuplot on the localhost.
```

*return* Integer

- Start `ox_sm1_gnuplot` on the localhost. It returns the descriptor of `ox_sm1_gnuplot`.
- Set `Xm_noX = 1` to start `ox_sm1_gnuplot` without a debug window.
- The descriptor is stored in `Gnuplot_proc`.

```
P = gnuplot.start();
```

#### Reference

`ox_launch`, `gnuplot`

### 1.1.2 gnuplot

`gnuplot.gnuplot(s|proc=p)`

:: Ask GNUPLOT to execute the command string *s*.

*return*      Void

*p*            Number

*s*            String

- The server executes the gnuplot command *s*. When an error occurs, the gnuplot itself terminates and ox\_sml\_gnuplot server automatically restarts gnuplot.
- gnuplot does not accept a long polynomial.
- gnuplot does not accept  $\wedge$ . Use `**` instead.

```
[232] P = gnuplot.start();
0
*Plot 3 dimensional graph.
[233] gnuplot.gnuplot("splot x**2-y**2;"|proc=P);
0
*Plot 2 dimensional graph.
[234] gnuplot.gnuplot("plot [-pi:pi] [-2:2] cos(x);");
0
*Output a graph as a postscript figure.
[235] gnuplot.output(|file="hoge.eps");
0
[236] gnuplot.gnuplot("plot sin(x)*cos(x);");
0
[237] gnuplot.gnuplot(|file="x11");
0

*Plot 3 dimensional graph hiding unvisible lines.
[236] gnuplot.gnuplot("set hidden3d");
0
[237] gnuplot.gnuplot("splot (x**2+y**2)*sin(x**2+y**2)");
0
[238] gnuplot.gnuplot("set isosamples 50");
0
[239] gnuplot.gnuplot("splot (x**2+y**2)*sin(x**2+y**2)");
```

#### Reference

`ox_launch`, `gnuplot.start`, `rtostr`, `gnuplot.plot_dots`

#### Reference Book

Yabuki Michiro, Otake Tuyoshi; Tukai konasu GNUPLOT, Techno Press, in Japanese, ISBN4-924998-11-7

### 1.1.3 gnuplot.plot\_dots

`gnuplot.plot_dots(d,s|proc=p)`

:: Plot the dots *d* with the style *s*.

*return*      Void

*p*            Number

*d*            List

*s*            String or 0

- Plot the dots *d* with the style *s*. *s* is a string of the form "style color point". Here, style can be lines, points, linespoints, impulses, dots, steps, errorbars, boxes, boxerrorbars. color can be 1 (red), 2 (green), 3 (blue), 4, ... , 8. point can be a number from 1 to 8. The color and point field can be omitted.
- When *d* == [ ], the screen will be cleared.

```
[239] P = gnuplot.start();
0
[240] gnuplot.plot_dots([ ],0);
0
[241] for (I=0; I<10; I++) gnuplot.plot_dots([[I,I^2]], " lines ");
[242] A = [ ];
[]
[243] for (I=0; I<10; I++) A = append(A, [ [I,I^2]]);
[244] A;
[[0,0],[1,1],[2,4],[3,9],[4,16],[5,25],[6,36],[7,49],[8,64],[9,81]]
[245] gnuplot.plot_dots(A, " lines ");
0
```

#### Reference

gnuplot.start, plot "fileName" with options(GNUPLOT command),  
gnuplot.clean, gnuplot

### 1.1.4 gnuplot.heat

**gnuplot.heat(dt,step)**

:: It solves the heat equation numerical and plots solutions

*return*      Void

*dt*          floating point number

*step*        Integer

- It solves the heat equation  $du/dt = d^2 u/dx^2$ ,  $u(t,0) = u(t,1) = 0$  with the initial condition  $u(0,x) = x$  ( $0 \leq x \leq 0.5$ ),  $u(0,x) = 1-x$  ( $0.5 \leq x \leq 1.0$ ).
- Heat\_N is the number of the meshes in the space.
- This function will be called **pde\_heat\_demo** in a future.

Algorithm: NOT Written. (Difference scheme. Courant-Levi-Friedrichs conditions.)

```
[232] gnuplot.set_heat_N(20)$
[233] gnuplot.heat(0.001,30)$
```

### 1.1.5 gnuplot.output

`gnuplot.output(|file=s)`

:: ask GNUPLOT to output graphic to the file *s* in the Postscript format.

*return*      Void

*s*            String

- ask GNUPLOT to output graphic to the file *s* in the Postscript format.
- When *s* is "x11" or this function is called without the argument, the output will be written to X11 display.

```
[273] gnuplot.output(|file="hoge.eps");
```

Graphic output of GNUPLOT will be written to hoge.eps as a Poscript file.

```
0
```

```
[274] gnuplot.gnuplot("plot tan(x)+sin(x);");
```

```
0
```

```
[275] gnuplot.output();
```

Usage of `gnuplot.output`: `gnuplot.output(|file="string")`

`gnuplot.output(|file="x11")`

Output device is set to X11

Reference

`gnuplot`

### 1.1.6 gnuplot.plot\_function

`gnuplot.gnuplot(f |proc=p)`

:: ask the `gnuplot` server to draw a graph of *f*

*return*      Void

*p*            Number

*f*            Polynomial or a list of polynomials

- ask the `gnuplot` server to draw a graph of *f*

```
[290] gnuplot.plot_function((x+sin(x))^2);
```

```
0
```

```
[291] gnuplot.plot_function([x,x^2,x^3]);
```

```
0
```

Reference

`gnuplot.to_gnuplot_format`

### 1.1.7 gnuplot.stop

`gnuplot.stop()`

:: Stop the `gnuplot` and remove the temporary fifo file.

*return*      Void

*s*            String

- Stop the GNUPLOT and remove the temporary fifo file generated by the `mkfifo` system call under the temporary directory.

```
[273] gnuplot.stop()
```

Reference

```
gnuplot.start
```

### 1.1.8 `gnuplot.setenv`

```
gnuplot.setenv(key,value)
```

```
::
```

*return*      Void

*key*          String

*value*        Object

- The *key* takes the value either in "gnuplot.callingMethod" or "plot.gnuplotexec".

Use the old method to communicate with gnuplot (version 3).

This method does not use mkfifo, but we need a patched version of gnuplot.

```
[273] gnuplot.setenv("gnuplot.callingMethod",0);
```

```
[274] gnuplot.setenv("plot.gnuplotexec",getenv("OpenXM_HOME")+"/bin/gnuplot4ox");
```

Calling your own gnuplot binary.

```
[274] gnuplot.setenv("plot.gnuplotexec","/cygdrive/c/program files/gnuplot/pgnuplot
```

Reference

```
gnuplot.start
```

# Index

(Index is nonexistent)

(Index is nonexistent)

## Short Contents

1	GNUPLOT Functions .....	1
	Index.....	6



# Table of Contents

<b>1</b>	<b>GNUPLOT Functions</b>	<b>1</b>
1.1	Functions	1
1.1.1	<code>gnuplot.start</code>	1
1.1.2	<code>gnuplot</code>	2
1.1.3	<code>gnuplot.plot_dots</code>	2
1.1.4	<code>gnuplot.heat</code>	3
1.1.5	<code>gnuplot.output</code>	4
1.1.6	<code>gnuplot.plot_function</code>	4
1.1.7	<code>gnuplot.stop</code>	4
1.1.8	<code>gnuplot.setenv</code>	5
<b>Index</b>		<b>6</b>