continuous_sys_generator provides a graphical user interface (fig.1) to solve noisy system of ODE written in form:

$$\left[\begin{cases} \frac{dx_1}{dt} = f_1(x_1, x_2, ..., x_k, t) + x_1 + y_1 \\ \frac{dx_2}{dt} = f_2(x_1, x_2, ..., x_k, t) + x_2 + y_2 \\ .... \\ \frac{dx_k}{dt} = f_k(x_1, x_2, ..., x_k, t) + x_k + y_k \end{cases}\right] + \begin{bmatrix} \bar{x}_1 + \bar{y}_1 \\ \bar{x}_2 + \bar{y}_2 \\ .... \\ \bar{x}_k + \bar{y}_k \end{bmatrix}$$

where: x_i , y_i - internal (multiplicative) Gaussian and white noises;

$\bar{x}_i$, $\bar{y}_i$ - external (additive) Gaussian and white noises.

If there is no noise influence in your system set all noise variances to zero. This means you have no noise.

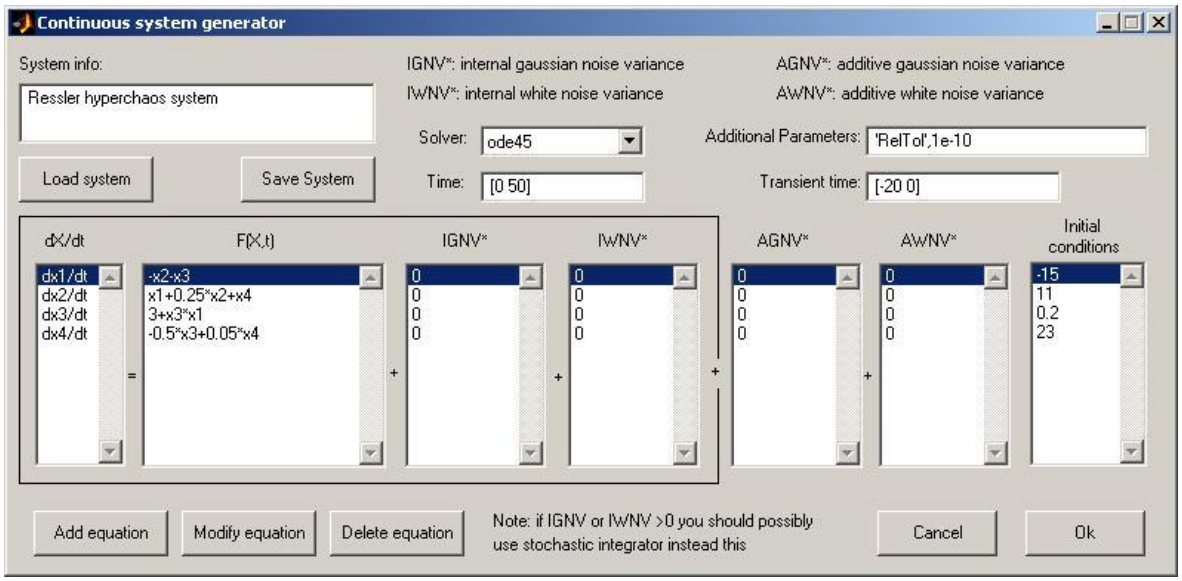

Fig.1 GUI

First of all system is solving at time interval taken from <Transient time:> field, with initial conditions listed in <Initial conditions> field. Then system is solving in time interval taken from <Time:> field with initial conditions that are last data values of a previous solution. These second solution returned to user. This order of operation may be usable if you doesn't need for well-defined initial conditions but need for the solution that lies on attractor. First column of returned matrix is time samples T, other columns are solutions Y. If you exactly know initial conditions that lie on attractor or want to look at a transient phenomenon leave the field <Transient steps:> empty.

Time fields may be noted in form: [init:step:final] or [init final] (ex.: [1:0.01:10] or [1 10]).

To run **continuous_sys_generator** type:

```
YourVariableName=continuous_sys_generator;
```

To run in silent (no-GUI) mode use following parameters:

```
data=continuous_sys_generator(Fx,ignv,iwnv,agnv,awnv,init,time,tr_time,so
lver,addit_param);
 solver - { 1 - ode45;
            2 - ode23;
            3 - ode113;
```

```
4 - ode15s;
5 - ode23s;
6 - ode23t;
7 - ode23tb; }
```

example: `data=continuous_sys_generator({'0.032*x1+0.5*x2-x3' 'x1-0.1*x2-x1^3' '0.1*x1'},{'0' '0' '0'},{'0' '0' '0'},{'0' '0' '0'},{'0' '0' '0'},{'0.01' '-0.01' '-0.02'}, '[0:0.02:50]', '[-10 0]',1,'');`

example for Duffing's equations:

```
data=continuous_sys_generator({'x2' '-0.1*x2-x1^3+11*cos(t)'}, {'0' '0'}, {'0' '0'}, {'0' '0'}, {'0' '0'}, {'0' '0'},'[0 50]','',1, '')
```

In the folder `systems/` you can find most known and popular chaotic systems studying in nonlinear dynamics.

continuous_sys_generator was made using Matlab 6.5, Windows platform.

This program is a part of Lab432 software for nonlinear analysis of time series (freely available by request)