Simulator

```
#
#
#
#
#
#
#
```

```
from casadi import *
from numpy import *
from pylab import *
```

We will investigate the working of Simulator with the help of the parametrically exited Duffing equation:

```

t = SX.sym('t')

u = SX.sym('u')
v = SX.sym('v')
states = vertcat(u,v)

eps   = SX.sym('eps')
mu    = SX.sym('mu')
alpha = SX.sym('alpha')
k     = SX.sym('k')
sigma = SX.sym('sigma')
Omega = 2 + eps*sigma

params = vertcat(eps,mu,alpha,k,sigma)
rhs    = vertcat(v,-u-eps*(2*mu*v+alpha*u**3+2*k*u*cos(Omega*t)))
```

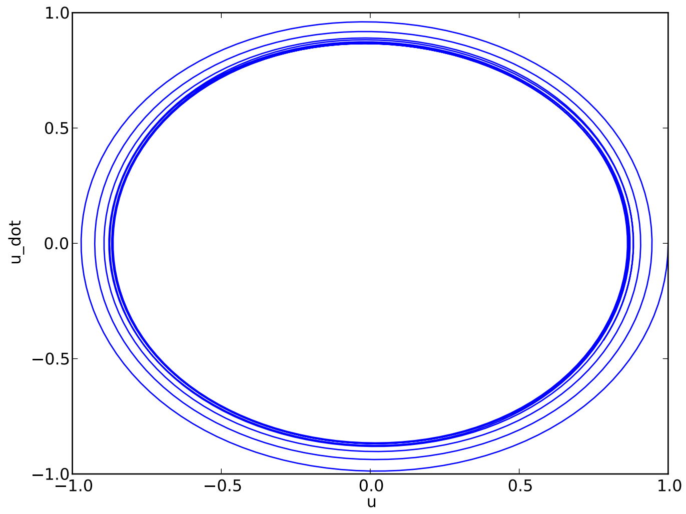
We will simulate over 50 seconds, 1000 timesteps.

```
dae={'x':states, 'p':params, 't':t, 'ode':rhs}
ts = linspace(0, 50, 1000)
integrator = integrator('integrator', 'cvodes', dae, {'grid':ts, 'output_t0':True})

sol = integrator(x0=[1,0], p=[0.1,0.1,0.1,0.3,0.1])
```

Plot the solution

```
plot(array(sol['xf'])[0,:], array(sol['xf'])[1,:])
xlabel('u')
ylabel('u_dot')
show()
```


u_dot
u