

nlin.pysim Simulating nonlinear systems in Python

Example nlin.pysim-1

re: a nonlinear unicycle

asdf

First, load some Python packages.

```
import numpy as np
import sympy as sp
from scipy.integrate import solve_ivp
import matplotlib.pyplot as plt
from IPython.display import display, Markdown, Latex
```

The state equation can be encoded via the following function f.

```
def f(t, x, u, c):
    dxdt = [
        x[3]*np.cos(x[2]),
        x[3]*np.sin(x[2]),
        x[4],
        1/c[0] * u(t)[0],
        1/c[1] * u(t)[1]
    ]
    return dxdt
```

The input function u must also be defined.

```
def u(t):
    return [
        15*(1+np.cos(t)),
        25*np.sin(3*t)
    ]
```

```
# %% Define time spans, initial values, and constants
tspan = np.linspace(0, 50, 300)
xinit = [0,0,0,0,0]
mass = 10
inertia = 10
c = [mass,inertia]

# %% Solve differential equation
sol = solve_ivp(
    lambda t, x: f(t, x, u, c),
    [tspan[0], tspan[-1]],
    xinit,
    t_eval=tspan
)
```

Let's first plot the trajectory and instantaneous velocity.

```
xp = sol.y[3]*np.cos(sol.y[2])
yp = sol.y[3]*np.sin(sol.y[2])
p = plt.figure();
plt.plot(sol.y[0],sol.y[1])
plt.quiver(sol.y[0],sol.y[1],xp,yp)
plt.xlabel('$$x$')
plt.ylabel('$$y$')
plt.show()
```

. Python code in this section was generated from a Jupyter notebook named horcrux.ipynb with a python3 kernel.

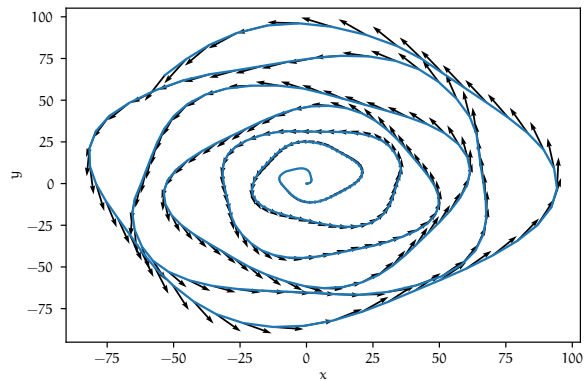


Figure pysim.1: