Steady State Curve

April 25, 2022

[ ]:
```
from Driver import Driver
from Race import Race
from TMEasy import Tire
from Vehicle import Vehicle
from Track import Track
import numpy as np
```

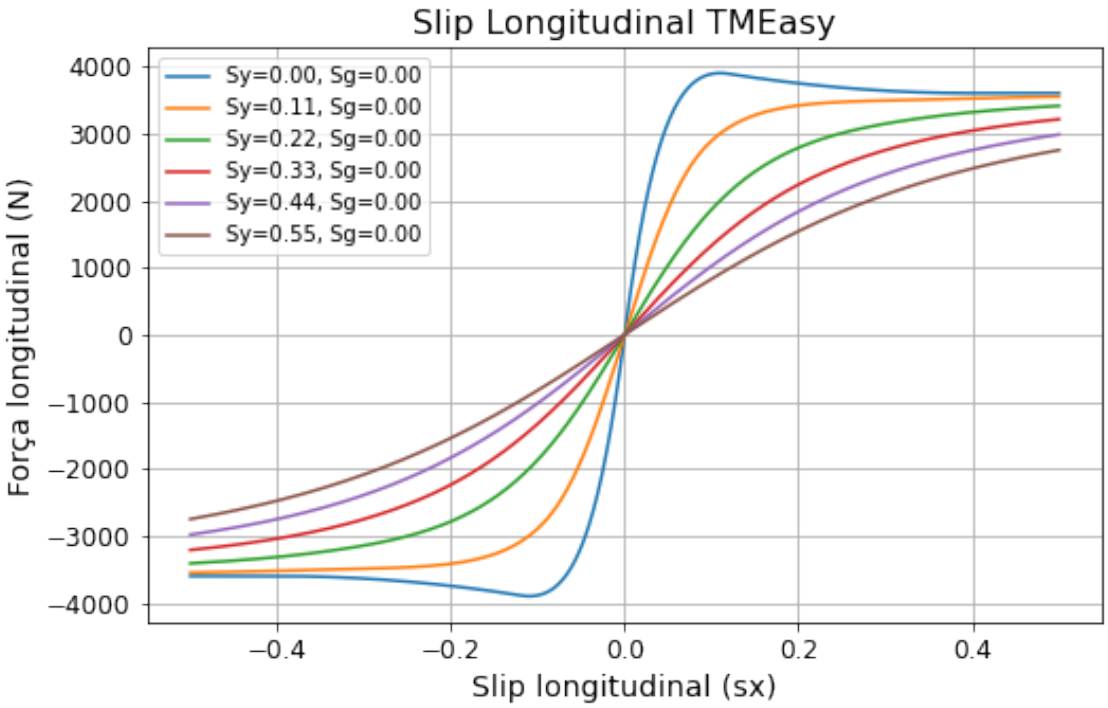
[ ]:
```
%load_ext autoreload
%autoreload 2
```

```
The autoreload extension is already loaded. To reload it, use:
  %reload_ext autoreload
```

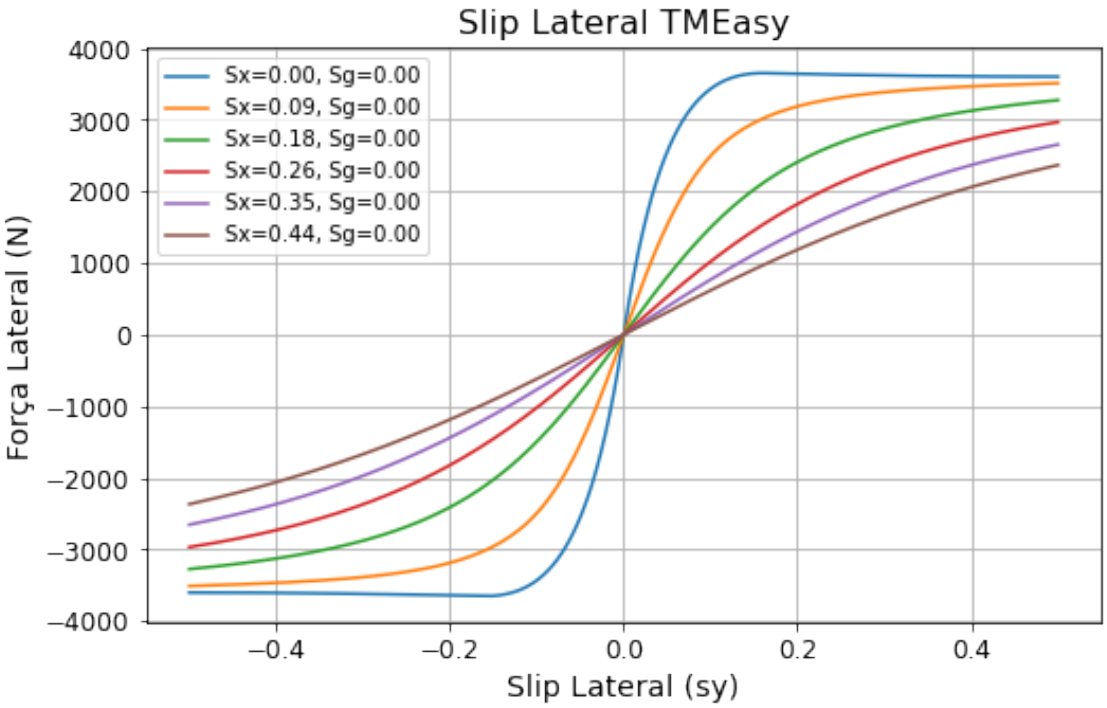
0.1 Tires

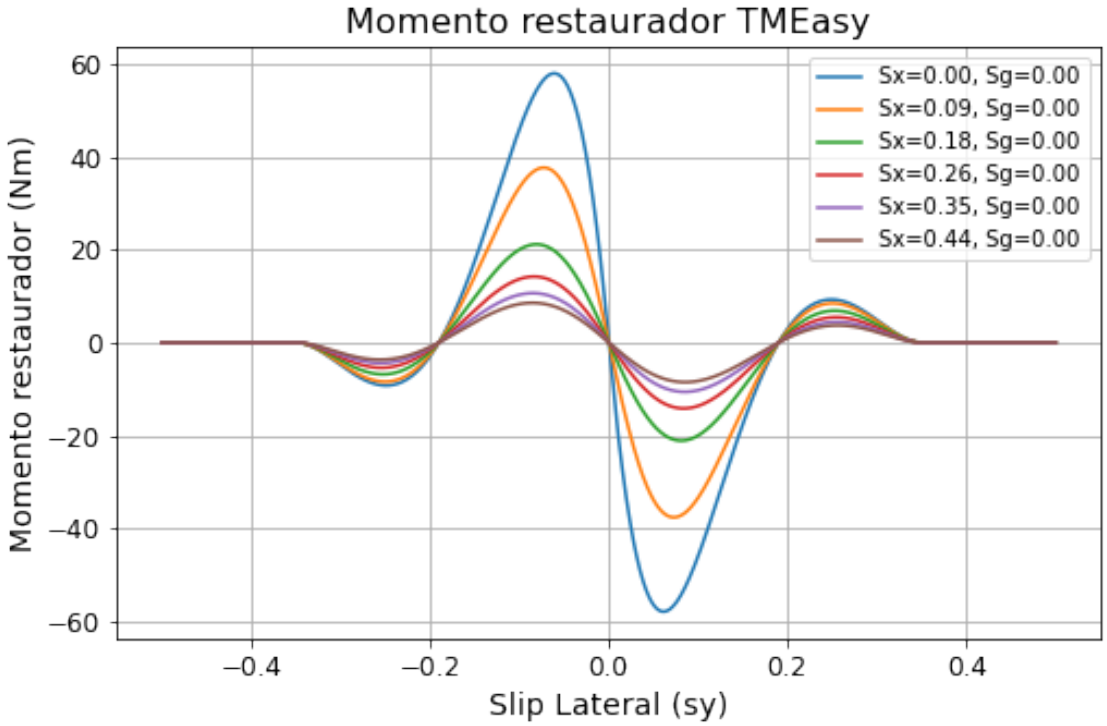
[ ]:
```
radius = 0.293
mass = 12
Jz_tire = 1
cz = 190000
dfx0 = 100000
dfy0 = 80000
fxm = 3900
fym = 3650
sxm = 0.11
sym = 0.16
fxs = 3600
fys = 3600
sxs = 0.4
sys = 0.5
n2L0 = 0.18
sy0 = 0.19
syE = 0.35
lamb = 2/3
frr = 0.015
TMEasy = Tire(radius, mass, Jz_tire, cz, dfx0, dfy0, fxm, fym, sxm, sym, fxs,␣
 ↪fys, sxs, sys, sy0, syE, lamb, n2L0, frr)
TMEasy.all_info()
```

Vertical force considered: 3500N
Camber slip = 0.00
Bore slip = 0.00



Coenering stiffness = 1186.9N/


Slip Lateral TMEasy
Sx=0.00, Sg=0.00
Sx=0.09, Sg=0.00
Sx=0.18, Sg=0.00
Sx=0.26, Sg=0.00
Sx=0.35, Sg=0.00
Sx=0.44, Sg=0.00
Força Lateral (N)
Slip Lateral (sy)
4000
3000
2000
1000
0
−1000
−2000
−3000
−4000
−0.4
−0.2
0.0
0.2
0.4


Momento restaurador TMEasy
Sx=0.00, Sg=0.00
Sx=0.09, Sg=0.00
Sx=0.18, Sg=0.00
Sx=0.26, Sg=0.00
Sx=0.35, Sg=0.00
Sx=0.44, Sg=0.00
Momento restaurador (Nm)
Slip Lateral (sy)
60
40
20
0
−20
−40
−60
−0.4
−0.2
0.0
0.2
0.4

0.2 Driver

[ ]:
```
driver = Driver(accelerator='PID', steering='steering')
# driver = Driver(steer='S')
```

```
steering defined.
```

0.3 Track

[ ]:
```
hx = 0
hy = 0
wx = 0.1
wy = 100
# mean = 0
# variance = 1

zf = lambda x, y: hx * np.sin(wx * x) + hy * np.sin(wy * y) #+ 0.01 * np.
 ↪sin(10*x)
vzf = lambda x, vx, y, vy: hx * wx * vx * np.cos(wx * x) + hy * wy * vy * np.
 ↪cos(wy * y)

track = Track(zf, vzf)
# track.set_noise(mean, variance)
track.profile(x=[0, 1000], y=[0,100], resolution=1)
```

0.4 Vehicle + Suspension

[ ]:
```
vehicle_mass = 1742
Ixx = 540
Iyy = 2398
Izz = 2617
lf = 1.07
lr = 1.605
wf = 1.517
wr = 1.505
af = 2.17
cd = 0.3
CG_height = 0.503
car = Vehicle(TMEasy, vehicle_mass, Ixx, Iyy, Izz, lf, lr, wf, wr, af, cd,␣
 ↪CG_height)
```

[ ]:
```
K_sf = 30800
K_sr = 28900
C_sf = 4500
C_sr = 3500
car.set_suspension(K_sf, K_sr, C_sf, C_sr)

G = 84e9
```

```
df = 0.022
af = 0.25
bf = 0.9
car.set_anti_roll_bar('f', df, af, bf, G)

dr = 0.013
ar = 0.3
br = 0.8
car.set_anti_roll_bar('r', dr, ar, br, G)

gamma1 = np.deg2rad(0) # O ângulo de cambagem positivo joga o carro para a␣
↪direita
gamma2 = np.deg2rad(0)
gamma3 = 0
gamma4 = 0
car.set_camber(gamma1, gamma2, gamma3, gamma4)
```

```
Anti-roll Bar (front) = 485.5 Nm/
Anti-roll Bar (rear)  = 36.5 Nm/
```

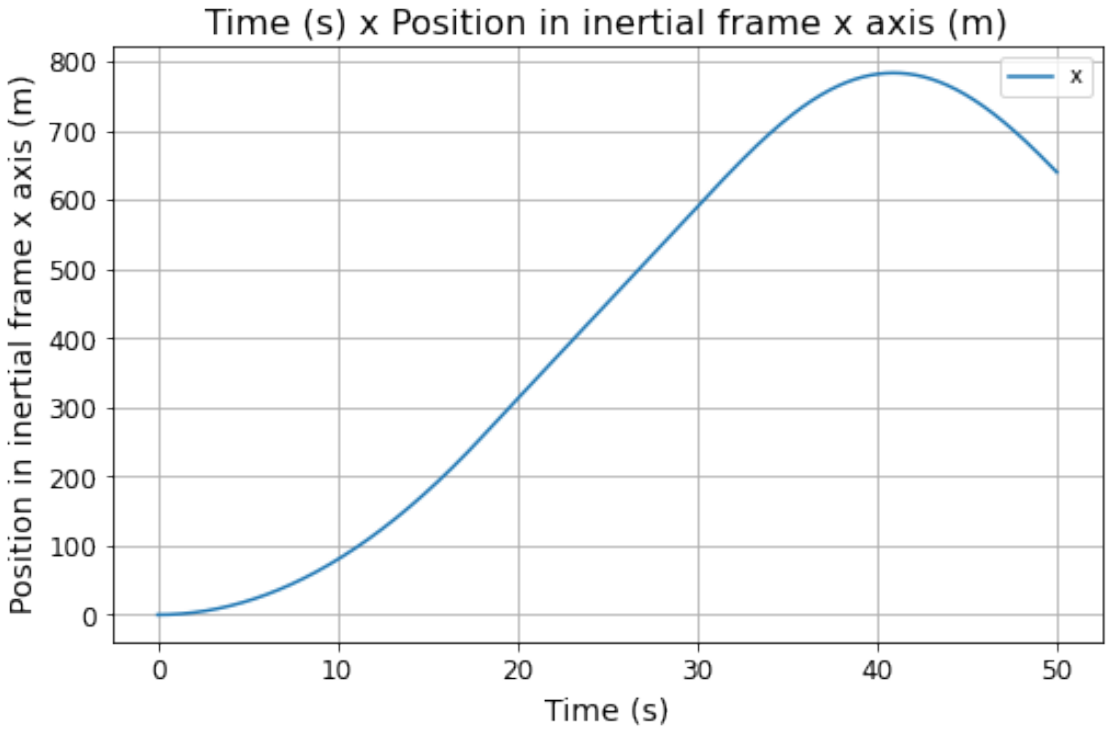
0.5 Simulation

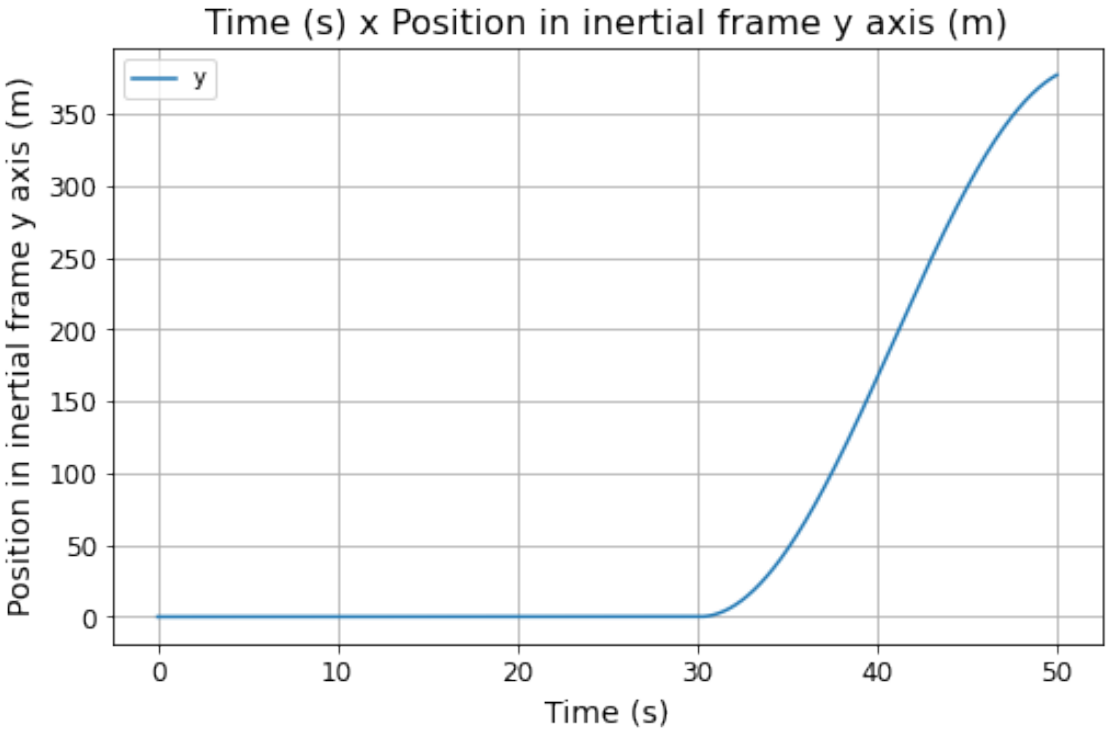
[ ]:
```
sim = Race(car, driver, track, maxTime=50, rtol=1e-3, atol=1e-3, maxStep=1e-2)
sim.post_process()
```

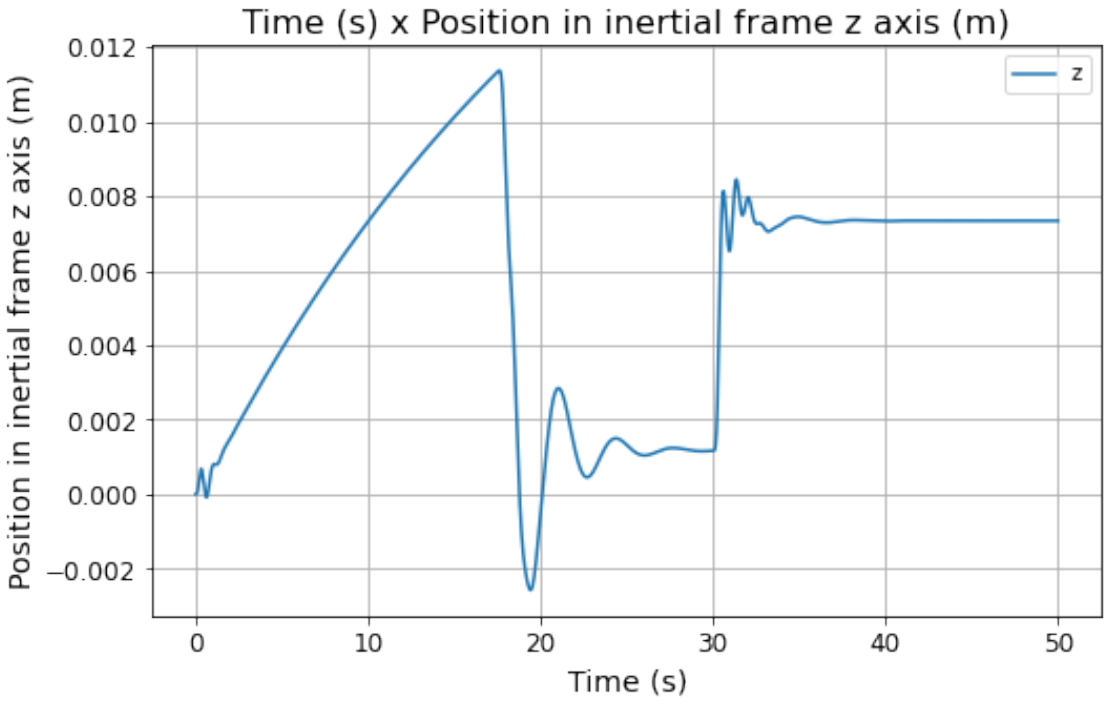
```
Solution Finished
```

0.6 Position

[ ]:
```
sim.x.plot2D()
sim.y.plot2D()
sim.z.plot2D()
```


Time (s) x Position in inertial frame x axis (m)
Position in inertial frame x axis (m)
Time (s)
x
0
100
200
300
400
500
600
700
800
0
10
20
30
40
50

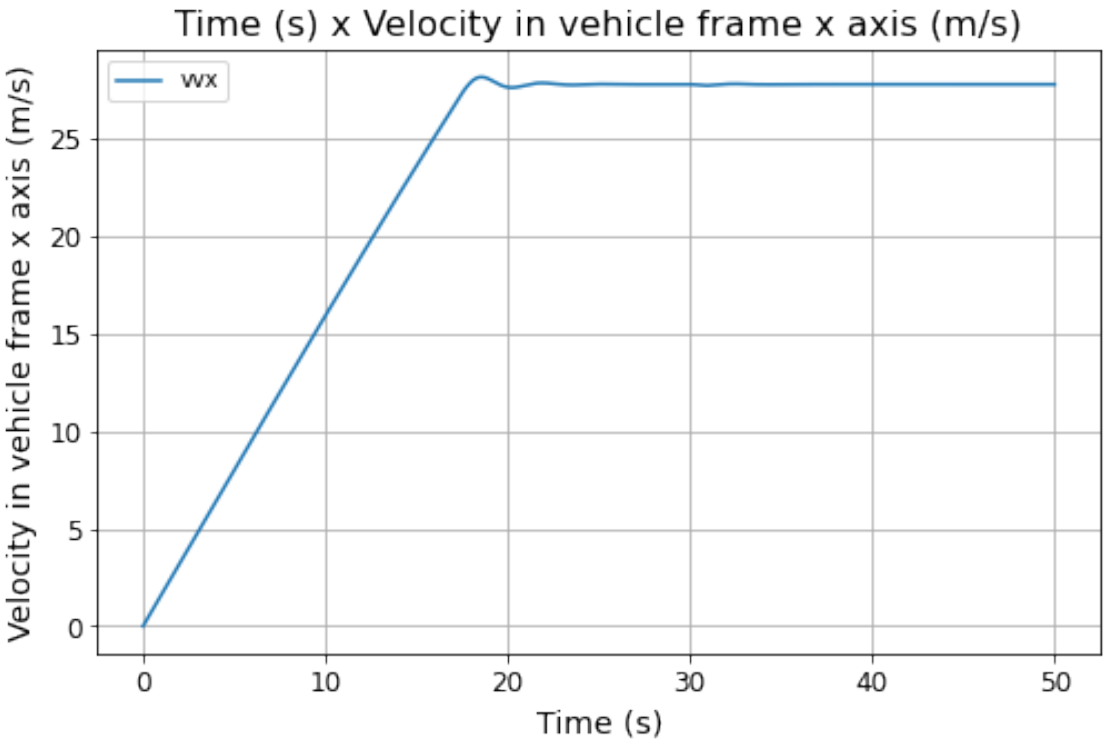

Time (s) x Position in inertial frame y axis (m)
Position in inertial frame y axis (m)
Time (s)
y
0
50
100
150
200
250
300
350
0
10
20
30
40
50

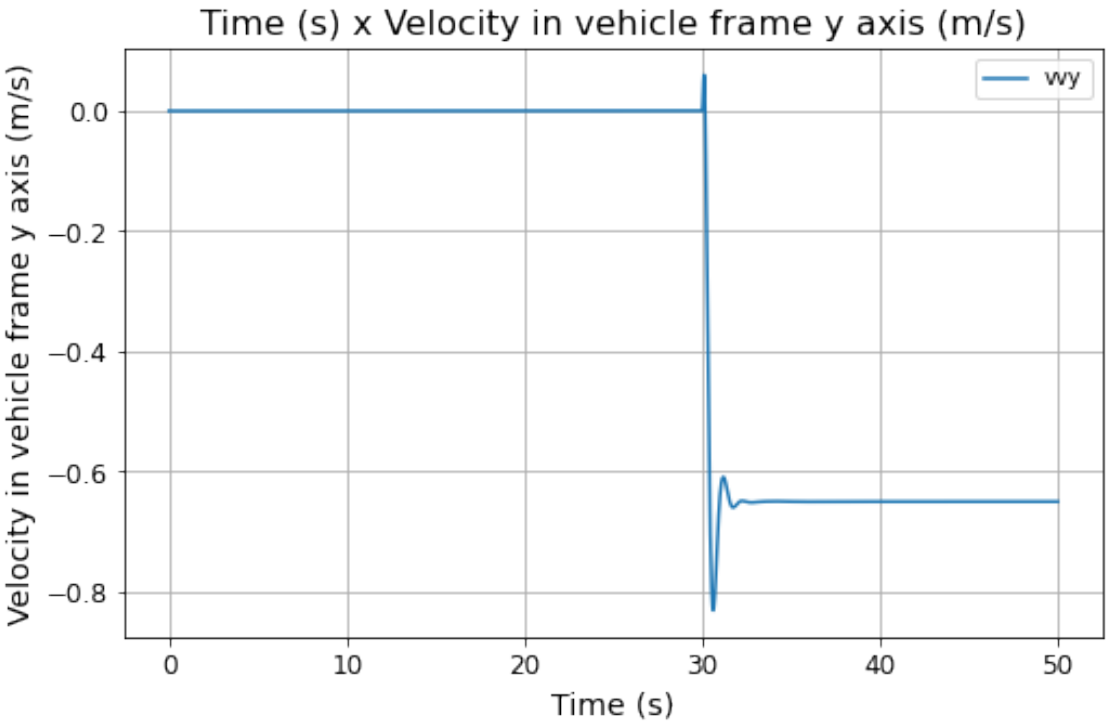


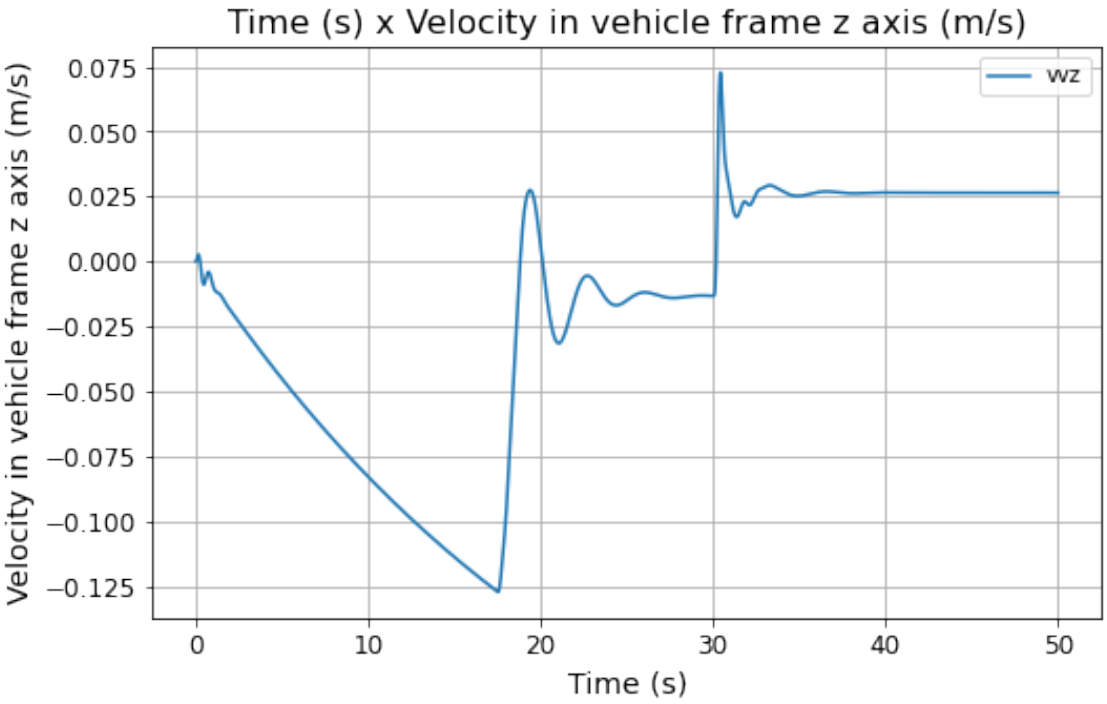
0.7 Speed

0.7.1 Vehicle frame

[ ]:
```
sim.vvx.plot2D()
sim.vvy.plot2D()
sim.vvz.plot2D()
```

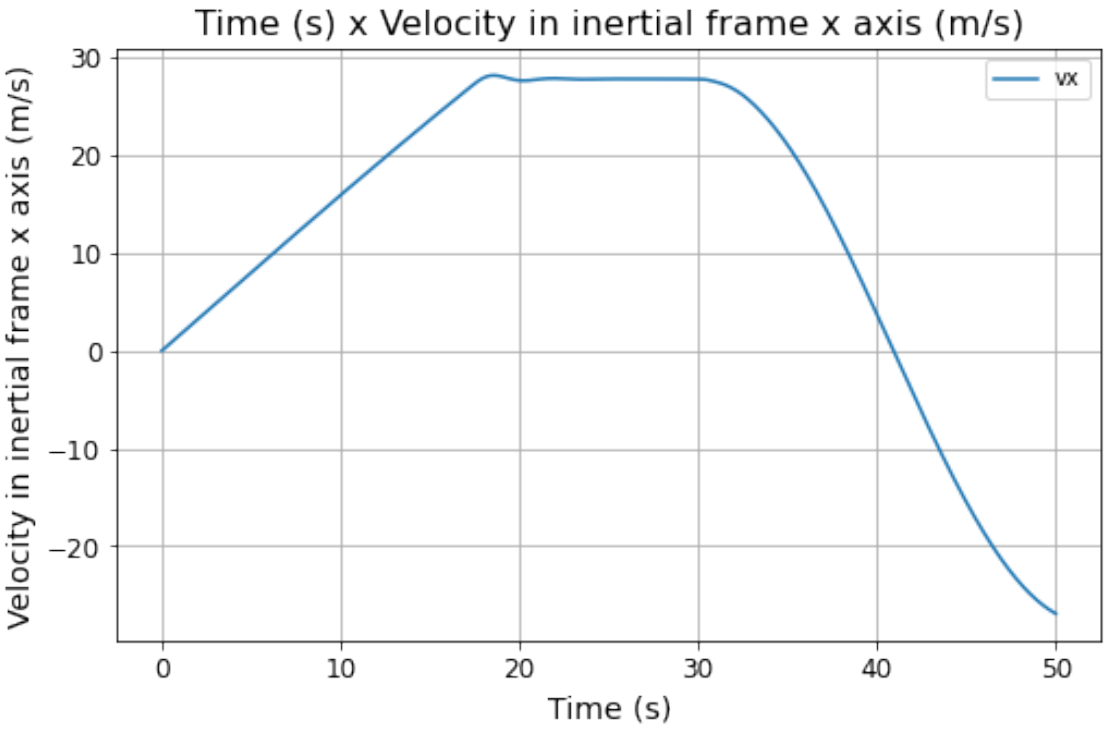

Time (s) x Velocity in vehicle frame x axis (m/s)
vx
Velocity in vehicle frame x axis (m/s)
0
5
10
15
20
25
0
10
20
30
40
50
Time (s)

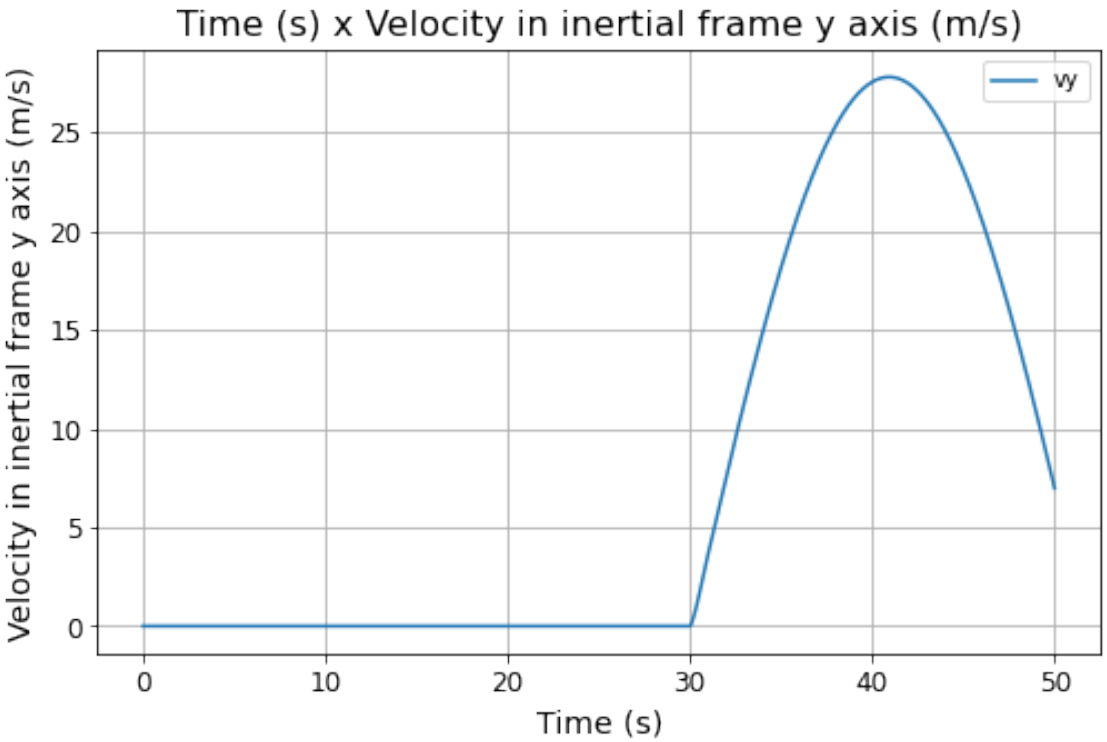

Time (s) x Velocity in vehicle frame y axis (m/s)
vy
Velocity in vehicle frame y axis (m/s)
0.0
−0.2
−0.4
−0.6
−0.8
0
10
20
30
40
50
Time (s)

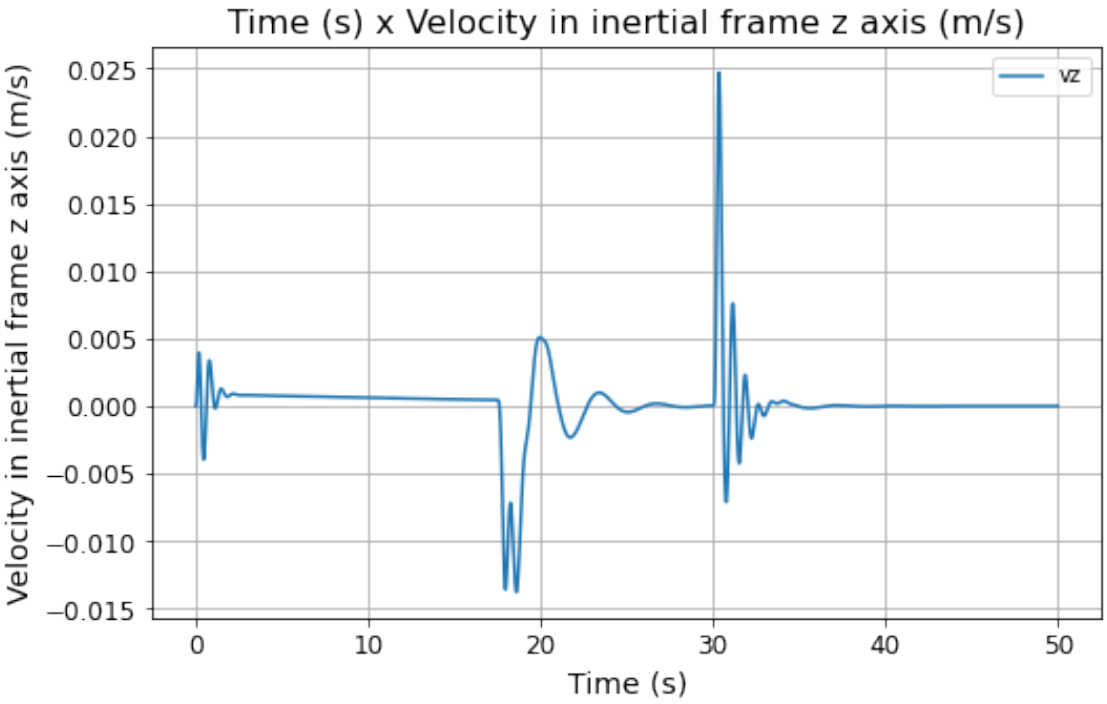


0.7.2 Inertial frame

[ ]:
```
sim.vx.plot2D()
sim.vy.plot2D()
sim.vz.plot2D()
```


Time (s) x Velocity in inertial frame x axis (m/s)
Velocity in inertial frame x axis (m/s)
Time (s)
vx
30
20
10
0
−10
−20
0
10
20
30
40
50

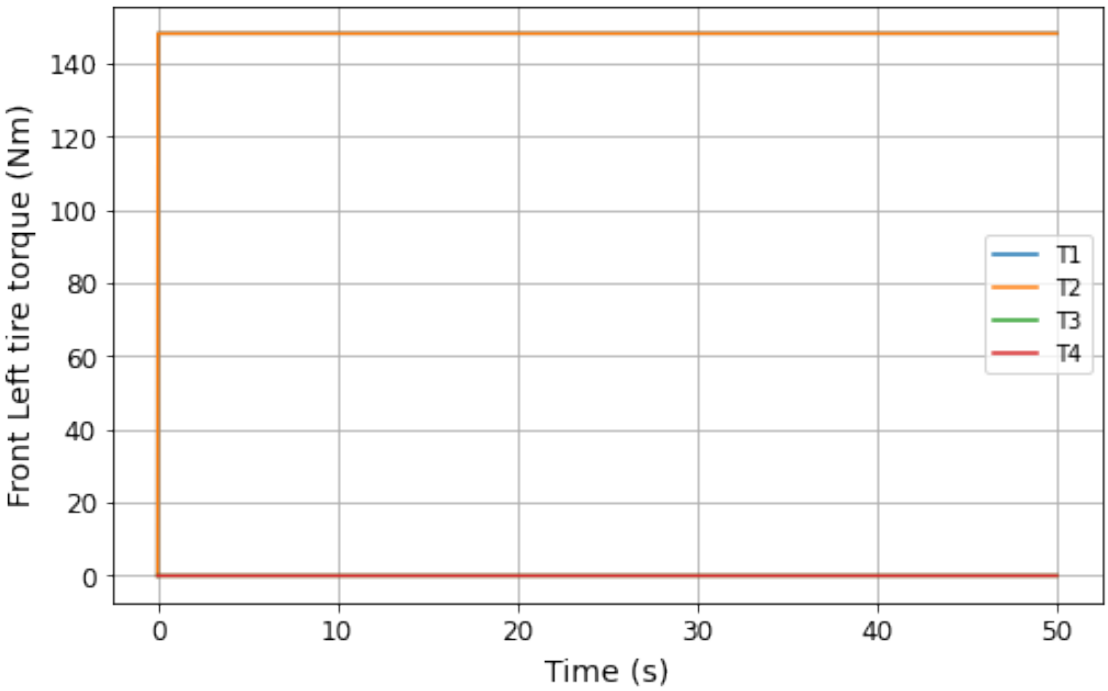

Time (s) x Velocity in inertial frame y axis (m/s)
Velocity in inertial frame y axis (m/s)
Time (s)
vy
25
20
15
10
5
0
0
10
20
30
40
50



0.8 Acceleration

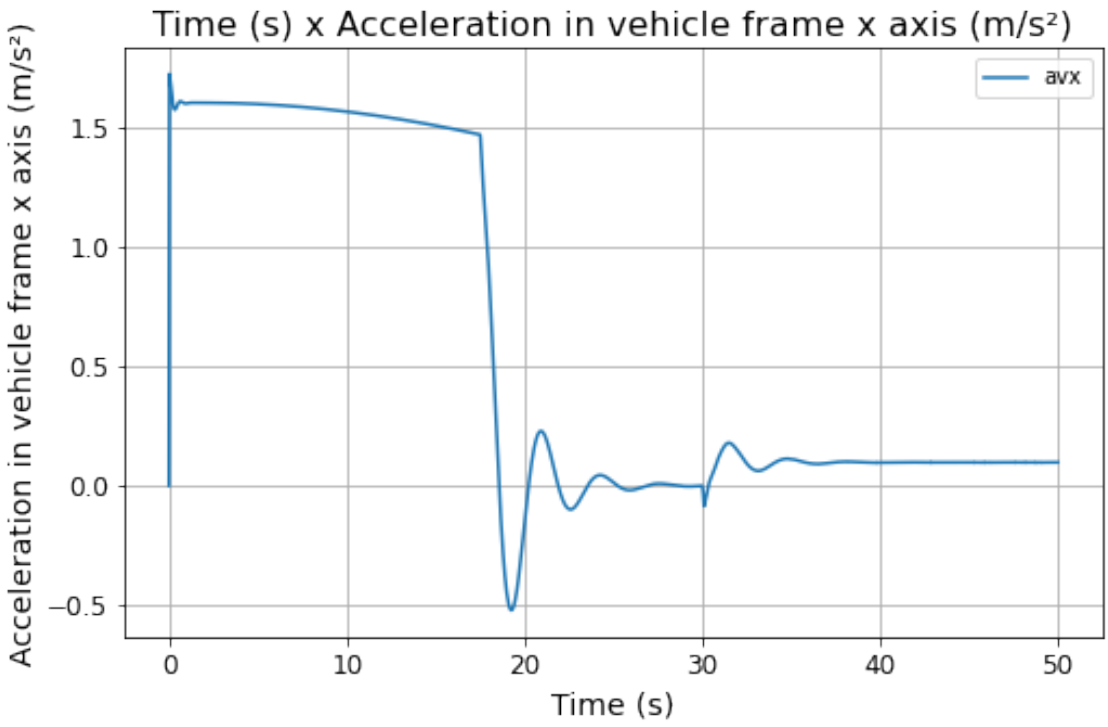
0.8.1 Input torque

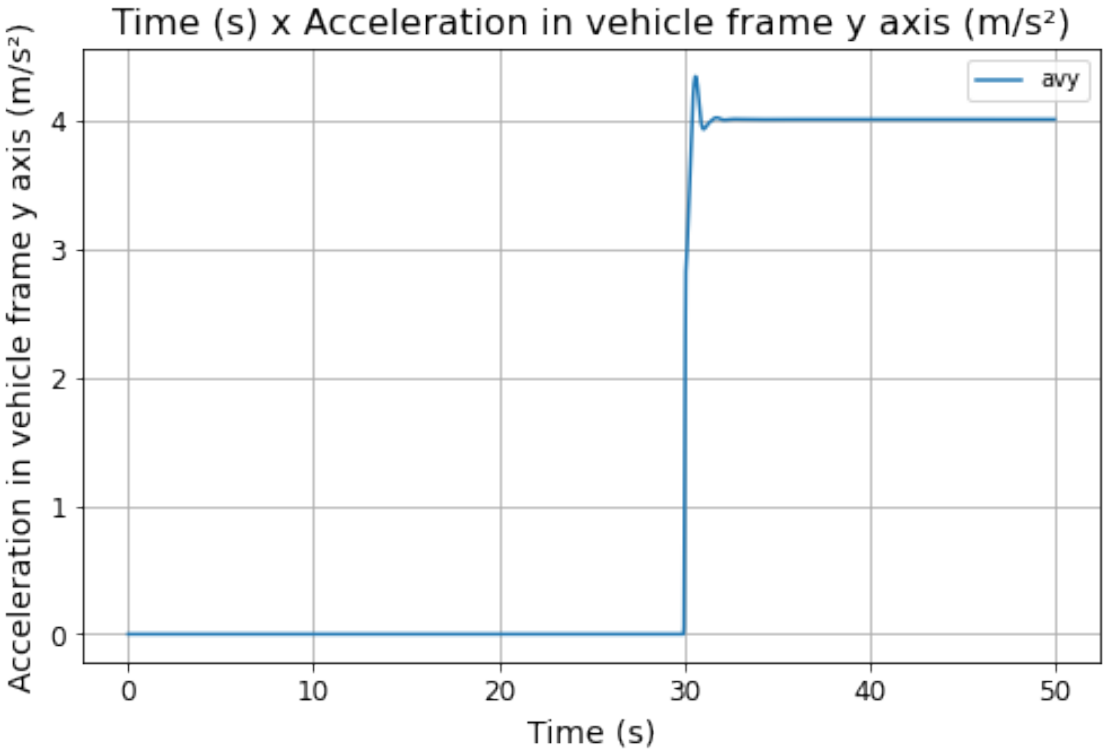
[ ]:
```
sim.T1.comparaNPlots([sim.T2, sim.T3, sim.T4])
```

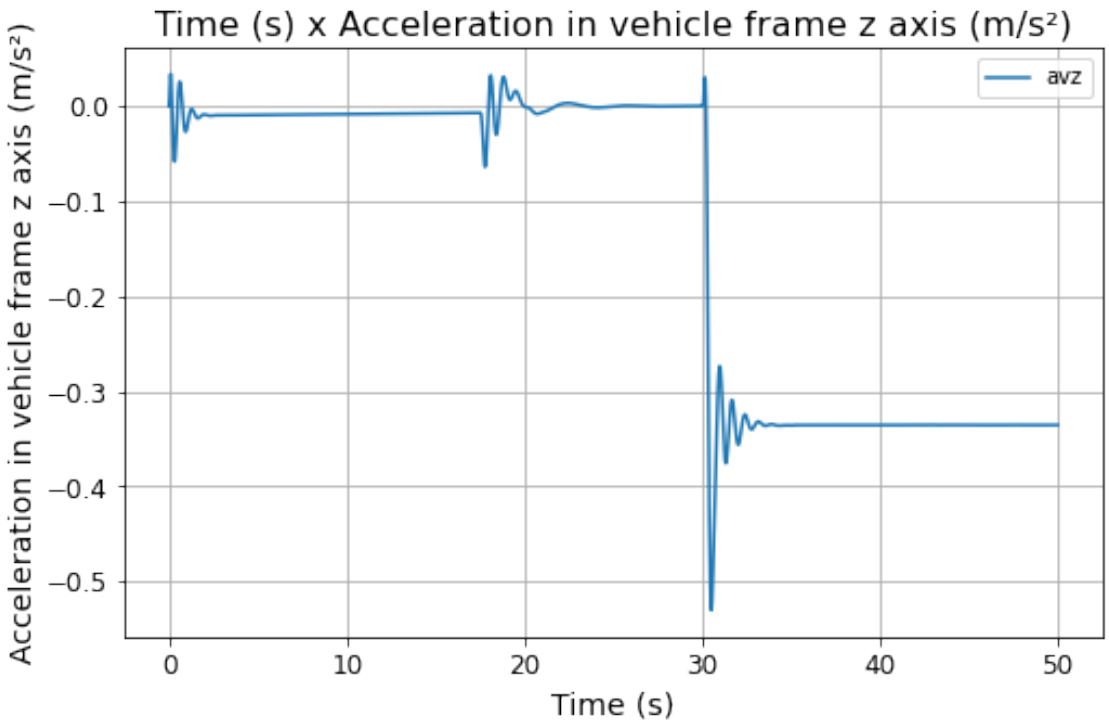


0.8.2 Vehicle Frame

[ ]:
```
sim.avx.plot2D()
sim.avy.plot2D()
sim.avz.plot2D()
```

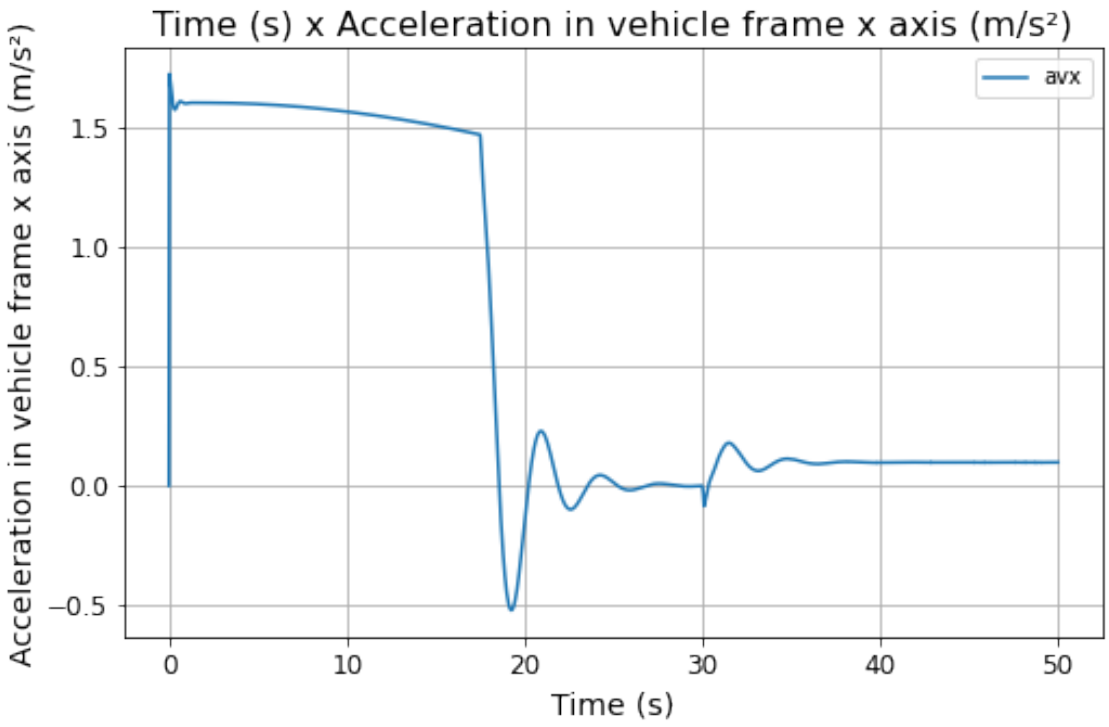

Time (s) x Acceleration in vehicle frame x axis (m/s²)
avx
Acceleration in vehicle frame x axis (m/s²)
1.5
1.0
0.5
0.0
−0.5
0
10
20
30
40
50
Time (s)

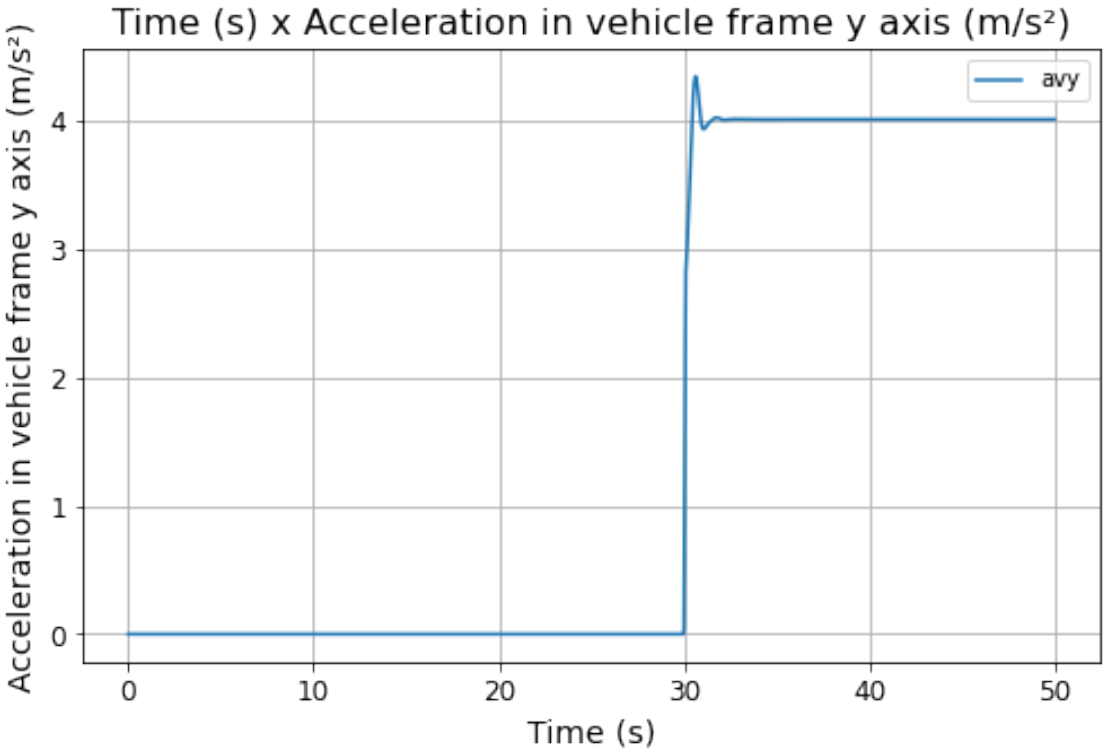

Time (s) x Acceleration in vehicle frame y axis (m/s²)
avy
Acceleration in vehicle frame y axis (m/s²)
4
3
2
1
0
0
10
20
30
40
50
Time (s)

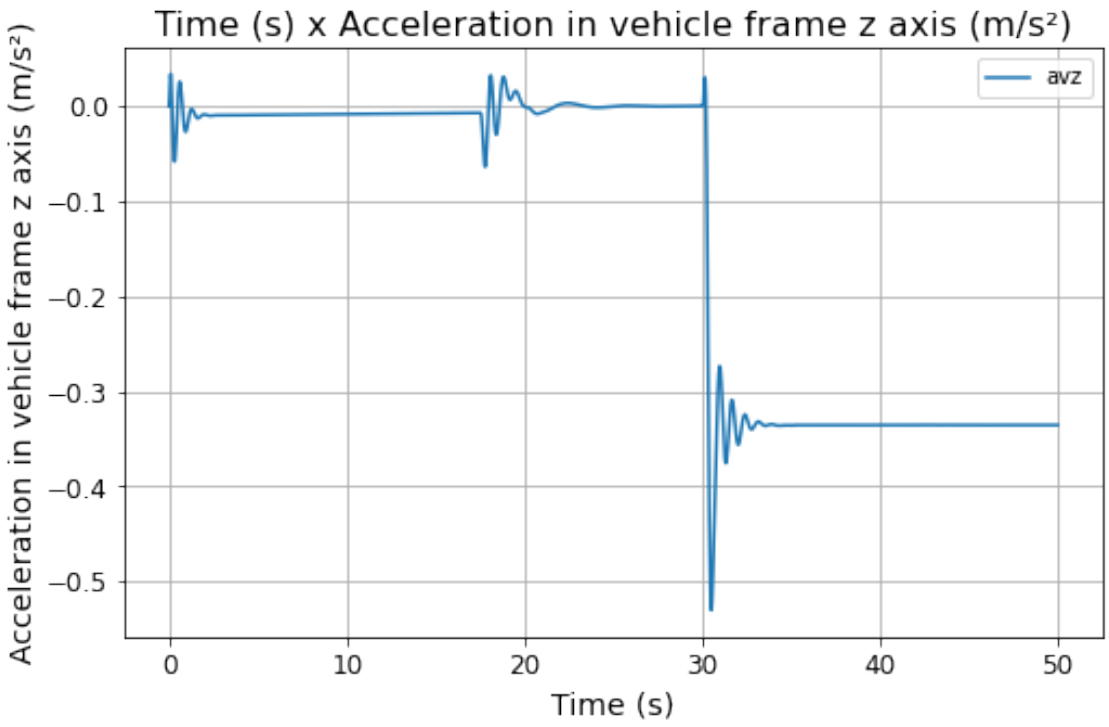


0.8.3 Inertial Frame

[ ]:
```
sim.avx.plot2D()
sim.avy.plot2D()
sim.avz.plot2D()
```


Time (s) x Acceleration in vehicle frame x axis (m/s²)
avx
Acceleration in vehicle frame x axis (m/s²)
1.5
1.0
0.5
0.0
−0.5
0
10
20
30
40
50
Time (s)

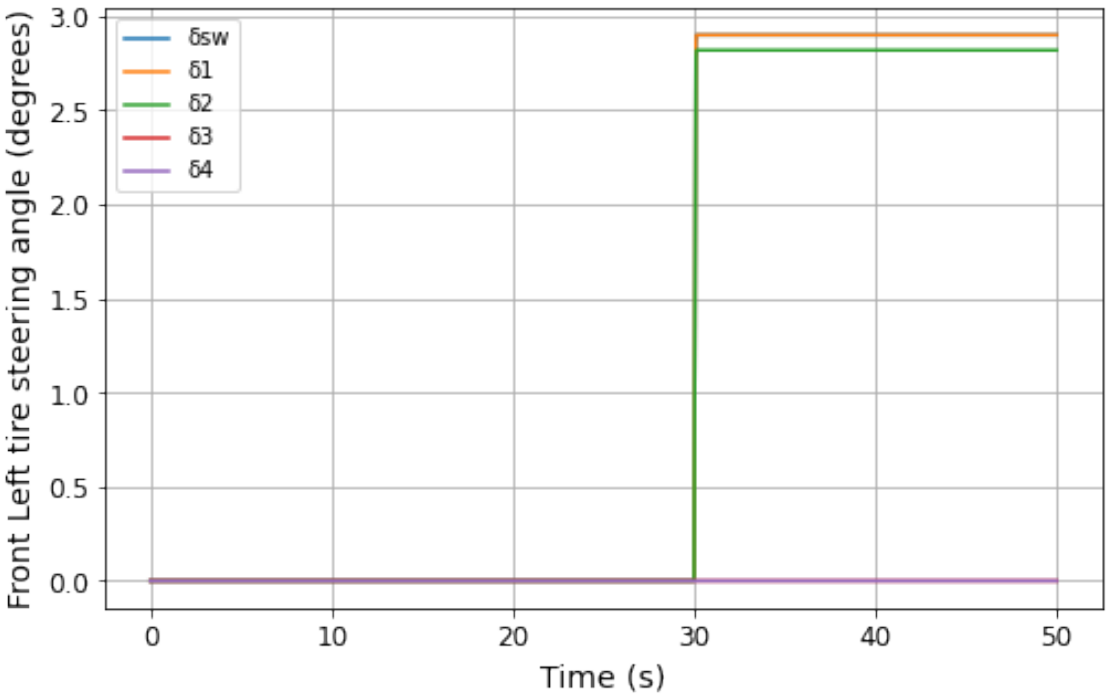

Time (s) x Acceleration in vehicle frame y axis (m/s²)
avy
Acceleration in vehicle frame y axis (m/s²)
4
3
2
1
0
0
10
20
30
40
50
Time (s)



0.9 Tires

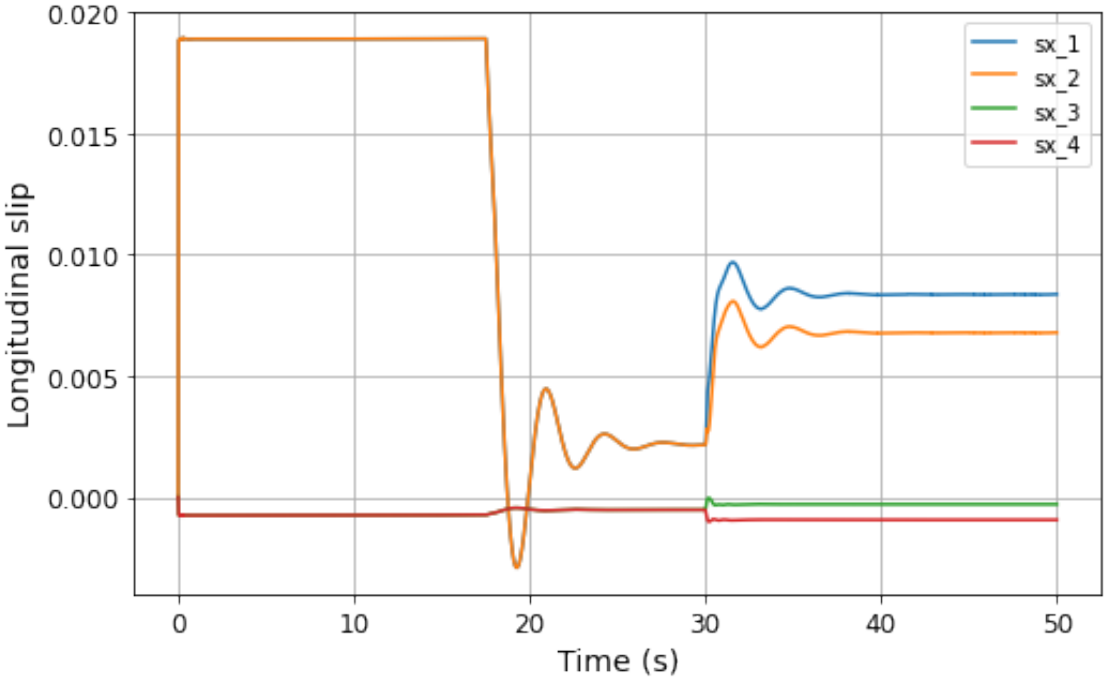
0.9.1 Delta

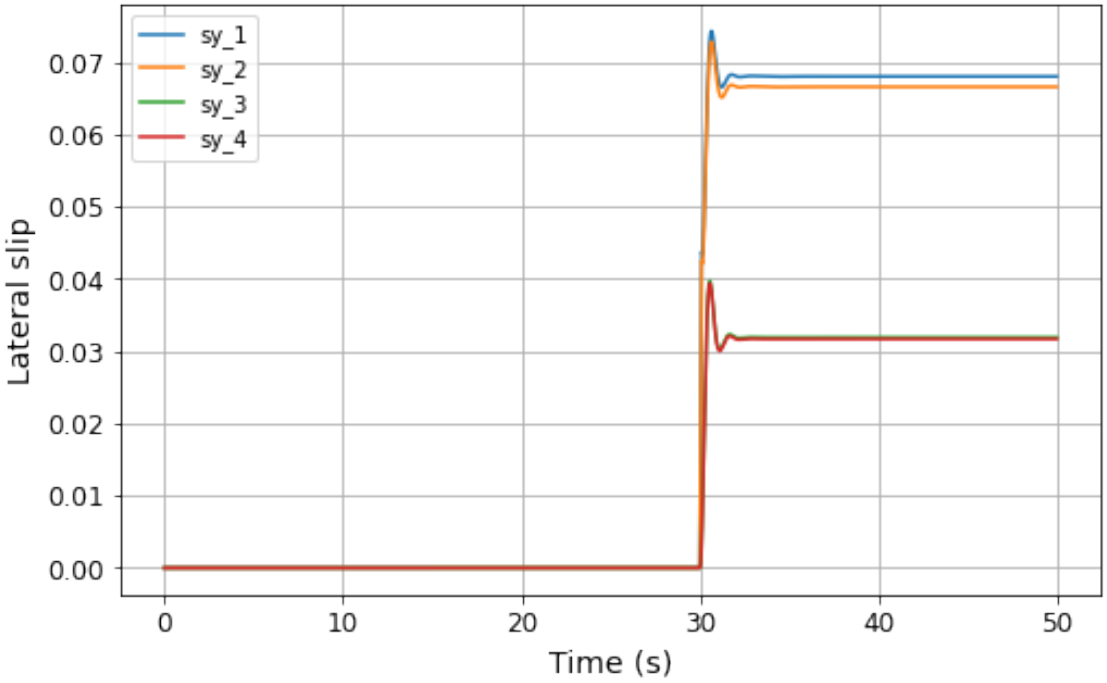
[ ]:
```
sim.delta_sw.comparaNPlots([sim.delta_1, sim.delta_2, sim.delta_3, sim.delta_4])
```



0.9.2 Slips

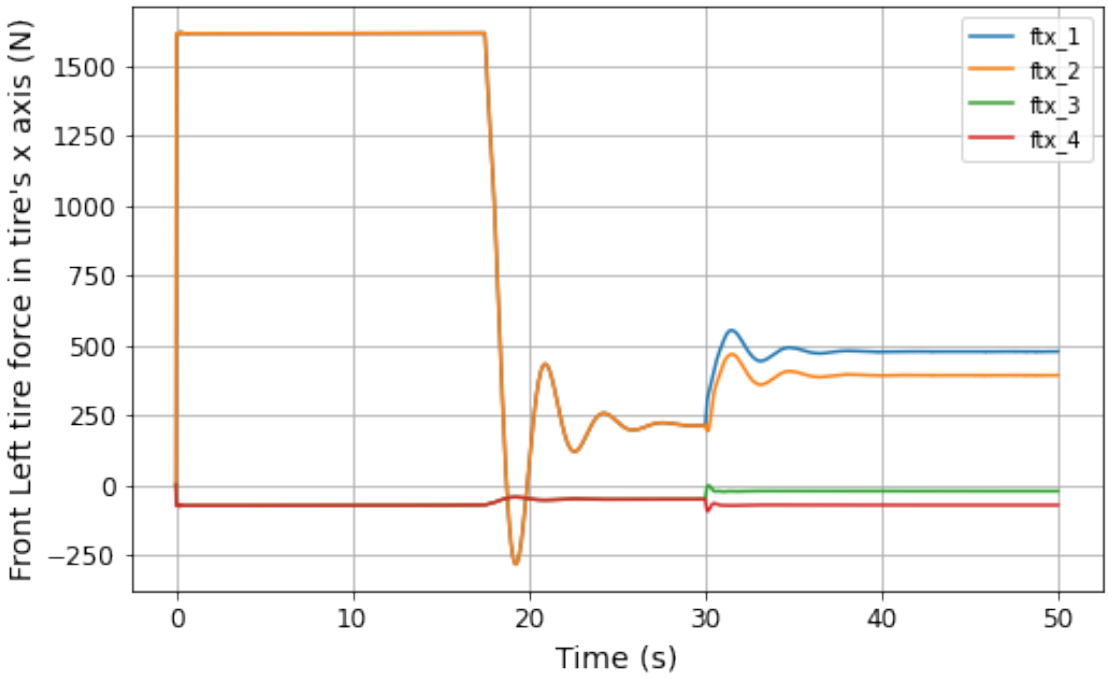
```
[ ]: sim.sx_1.comparaNPlots([sim.sx_2, sim.sx_3, sim.sx_4])
     sim.sy_1.comparaNPlots([sim.sy_2, sim.sy_3, sim.sy_4])
```

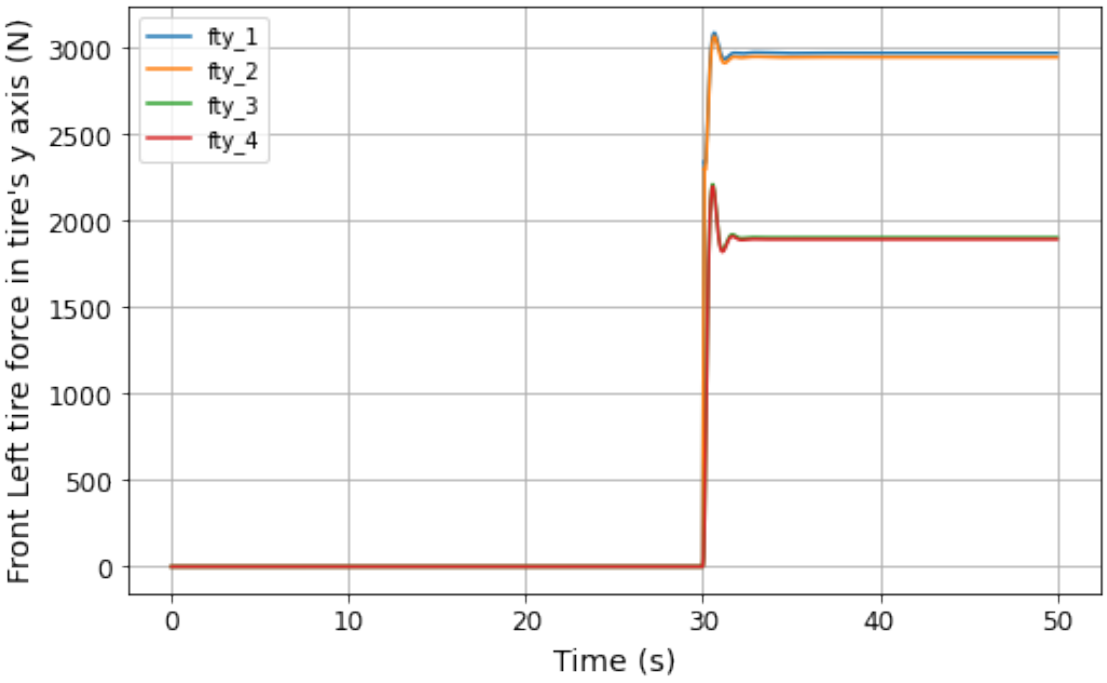


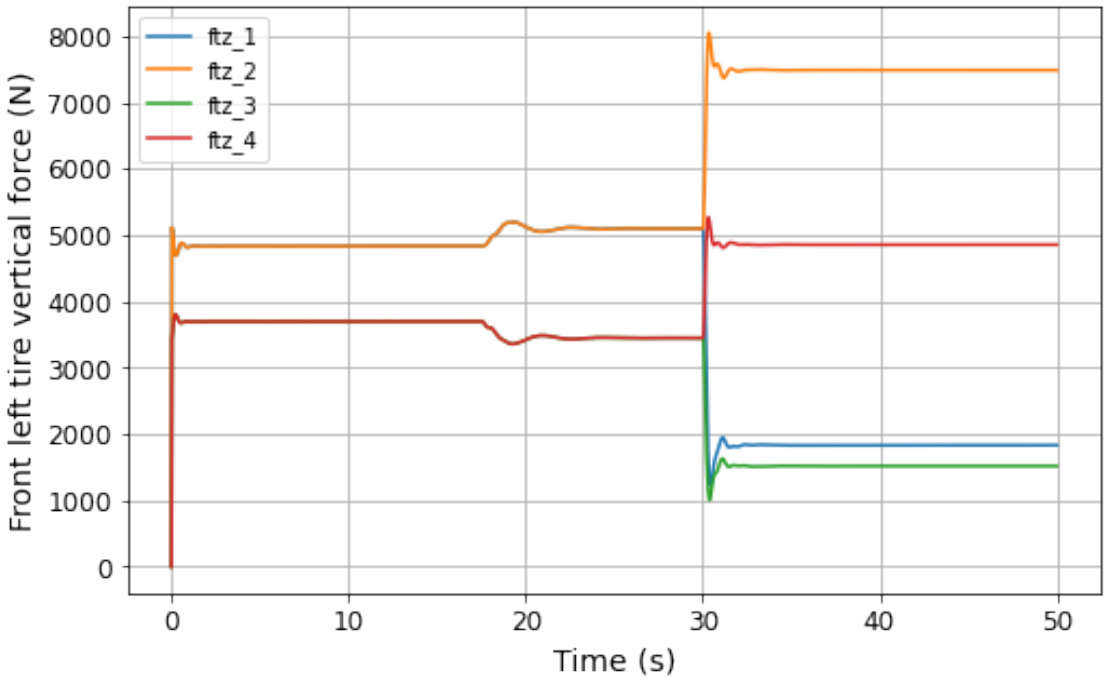


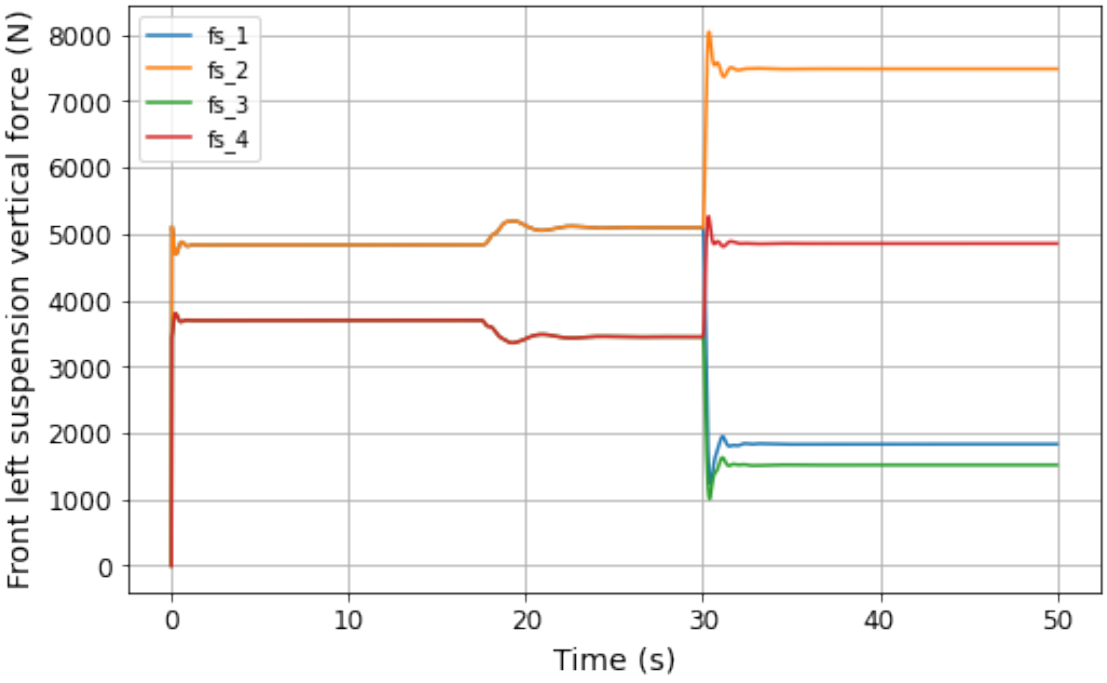
0.9.3 Forces

[ ]:
```
sim.ftx_1.comparaNPlots([sim.ftx_2, sim.ftx_3, sim.ftx_4])
sim.fty_1.comparaNPlots([sim.fty_2, sim.fty_3, sim.fty_4])
sim.ftz_1.comparaNPlots([sim.ftz_2, sim.ftz_3, sim.ftz_4])
sim.fs_1.comparaNPlots([sim.fs_2, sim.fs_3, sim.fs_4])
```


Front Left tire force in tire's x axis (N)
1500
1250
1000
750
500
250
0
−250
0
10
20
30
40
50
Time (s)
ftx_1
ftx_2
ftx_3
ftx_4

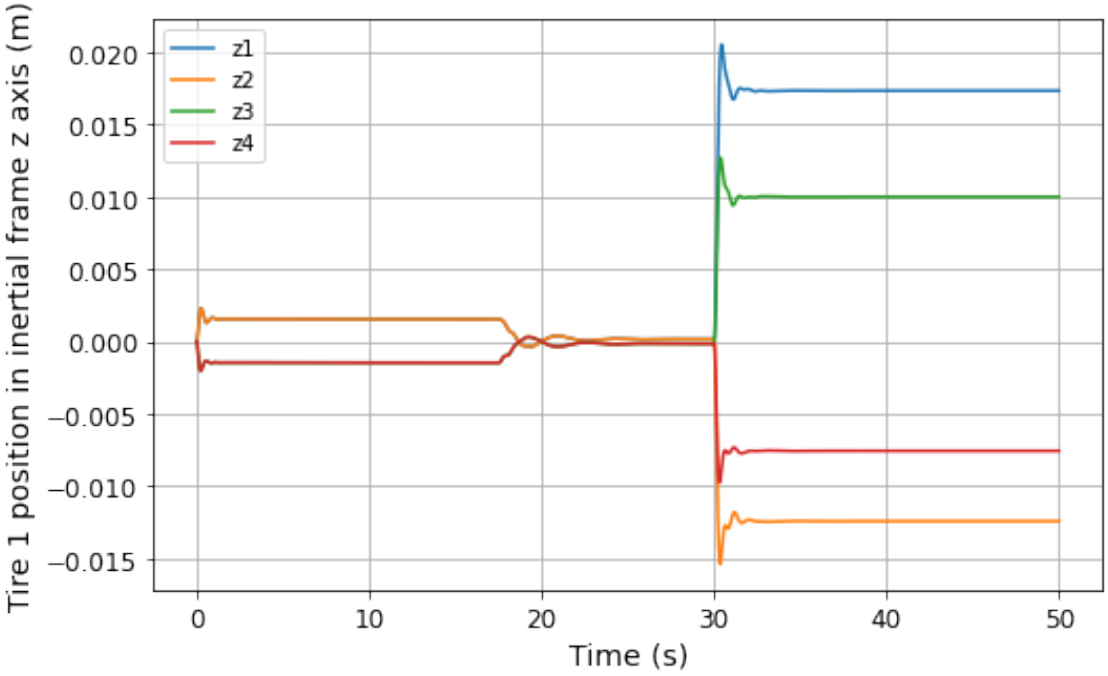

Front Left tire force in tire's y axis (N)
3000
2500
2000
1500
1000
500
0
0
10
20
30
40
50
Time (s)
fty_1
fty_2
fty_3
fty_4


ftz_1
ftz_2
ftz_3
ftz_4
Front left tire vertical force (N)
Time (s)
0
1000
2000
3000
4000
5000
6000
7000
8000
0
10
20
30
40
50


fs_1
fs_2
fs_3
fs_4
Front left suspension vertical force (N)
Time (s)
0
1000
2000
3000
4000
5000
6000
7000
8000
0
10
20
30
40
50

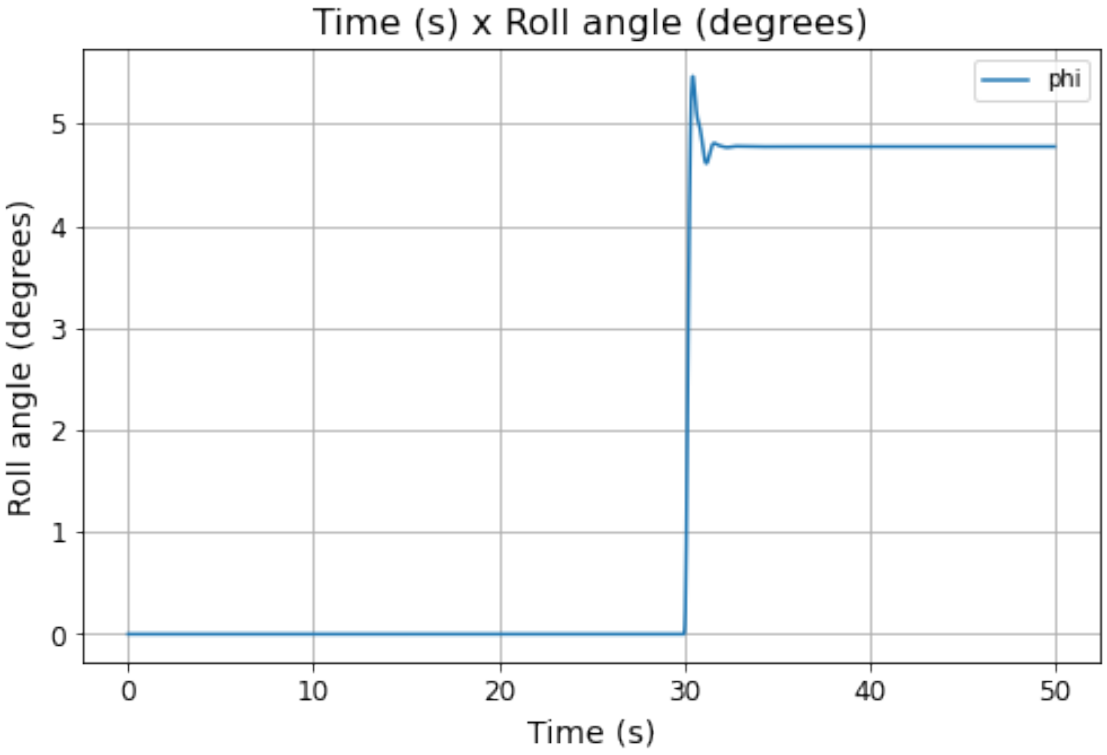
0.9.4 Tire height (z position)

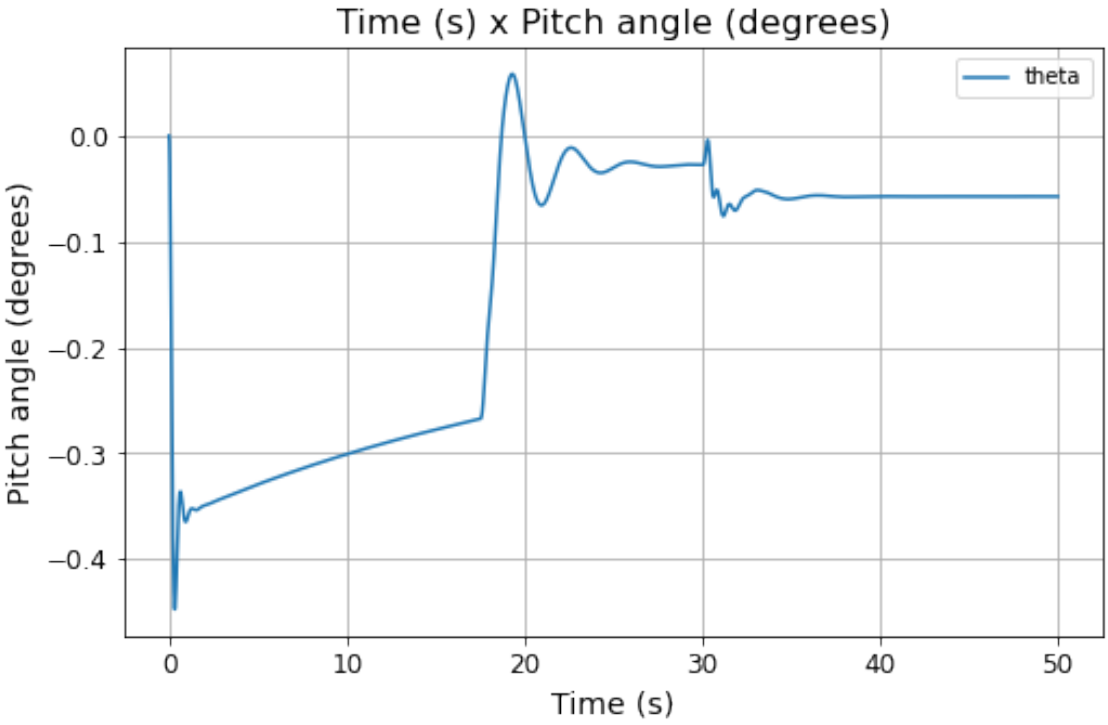
[ ]:
```
sim.z1.comparaNPlots([sim.z2, sim.z3, sim.z4])
```

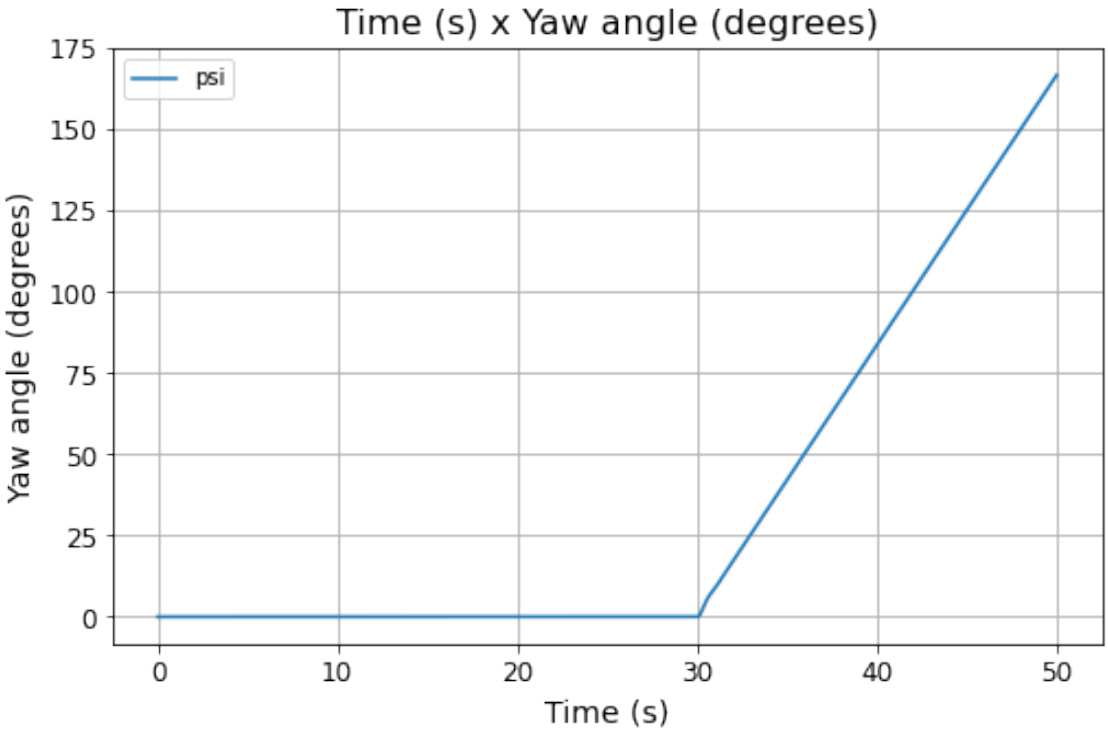


0.10 Angles

[ ]:
```
sim.phi.plot2D()
sim.theta.plot2D()
sim.psi.plot2D()
```

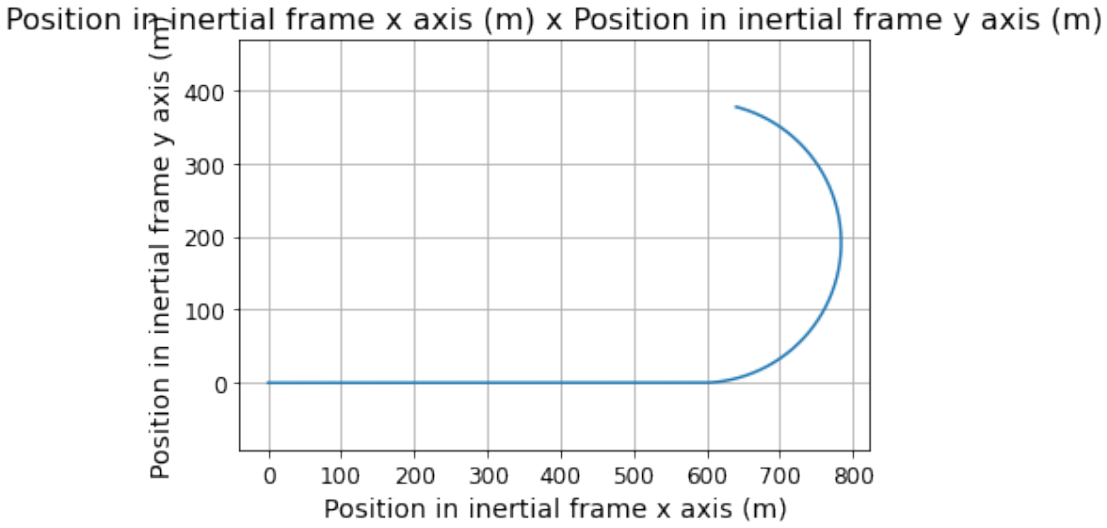

Time (s) x Roll angle (degrees)
phi
Roll angle (degrees)
0
1
2
3
4
5
0
10
20
30
40
50
Time (s)


Time (s) x Pitch angle (degrees)
theta
Pitch angle (degrees)
0.0
−0.1
−0.2
−0.3
−0.4
0
10
20
30
40
50
Time (s)



0.11 Parametric

[ ]:
```
sim.xy.plotparametric()
```



0.12 Animation

[ ]:
```
sim.animate(50, save=False)
```