1. Synaptic current

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
```

In [2]:
```
def get_timeline(t_max, dt):
    return np.arange(0, t_max, dt)
```

In [3]:
```
def membrane_euler(x_func, g_syn_func, v_func, i_func, params, initial_values):
    t = get_timeline(params['t_max'], params['dt'])

    v = np.zeros(len(t))
    g_syn = np.zeros(len(t))
    x = np.zeros(len(t))
    i_syn = np.zeros(len(t))

    v[0] = initial_values['v_0']
    g_syn[0] = initial_values['g_syn_0']
    x[0] = initial_values['x_0']

    for i in range(len(t)-1):
        x[i+1] = x[i] + x_func(x[i],t[i], params) * params['dt']
        g_syn[i+1] = g_syn[i] + g_syn_func(g_syn[i],x[i], t[i], params) * params['dt']
        v_new , i_syn[i] = v_func(v[i], g_syn[i], t[i], i_func, params)
        v[i+1] = v[i] + v_new * params['dt']
    return v, i_syn, g_syn, t
```

In [4]:
```
def passive_membrane_potential(v, g_syn, t, i_func, params) :
    a = (1/params['tau_m'])
    current = i_func(g_syn,v, t, params)
    c = params['e_m'] + params['r_m'] * current + params['Rm_Ie'] # Lets consider (e_m + r_m * i+ r_m * i_e) = c
    return a*(-v + c) , current
```

In [5]:
```
def i_syn_func(g_syn, v, t, params):
    return g_syn * (v - params['e_syn'])
```

In [6]:
```
def i_mem_func(v, params):
    return (v - params['e_m']) / params['r_m']
```

In [7]:
```
def g_syn(g_syn_1, x, t, params) :
    return (x - g_syn_1) / params['tau_syn']
```

In [8]:
```
def mem_x(x_1, t, params):
    d_x = -x_1 / params['tau_syn']

    if(np.abs(t - params['t_spike']) < (0.5 * params['dt'])) :
        d_x += params['g_*'] / params['dt']

    return d_x
```

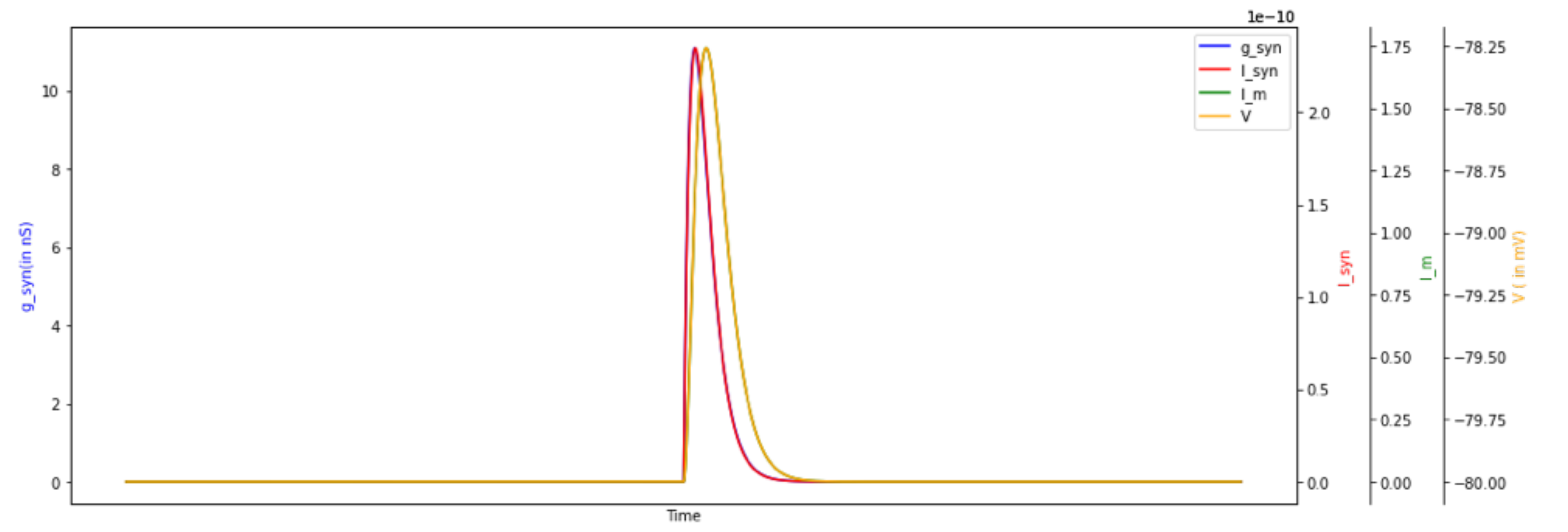
In [9]:
```
params = {
    'r_m':1e7, # in Ohm
    'e_m':-80e-3, # in Volt
    'tau_m':10e-3, # in seconds
    'Rm_Ie' : 0,
    'tau_syn':10e-3,
    'g_*' : 30e-9
}

initial_values = {
    'x_0': 0,
    'g_syn_0': 0,
    'v_0' : -80e-3 #v(0) = Em
}
```

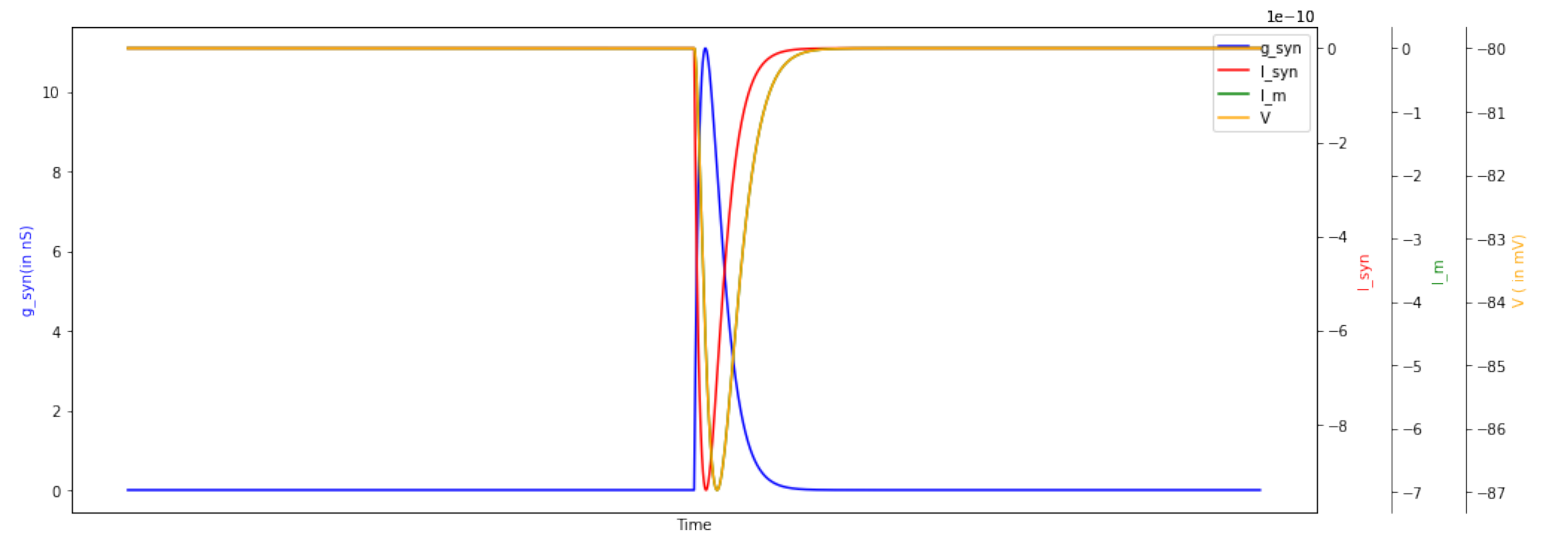
1a) Characterize the response to a single pre-synaptic spike

In [40]:
```
def plot_func(g_syn, i_syn, i_m, v, t):
    fig = plt.figure(figsize=(15, 6))
    host = fig.add_subplot(111)

    par1 = host.twinx()
    par2 = host.twinx()
    par3 = host.twinx()

#     host.set_xlim(0, 1)
#     host.set_ylim(0, 100)
#     par1.set_ylim(0, 400)
#     par2.set_ylim(1, 65)
#     par3.set_ylim(-80, 30)

    host.set_xlabel("Time")
    host.set_ylabel("g_syn(in nS)")
    par1.set_ylabel("I_syn")
    par2.set_ylabel("I_m")
    par3.set_ylabel("V ( in mV)")

    p1, = host.plot(t, g_syn*(10**9), color='blue',label="g_syn")
    p2, = par1.plot(t, i_syn, color='red', label="I_syn")
    p3, = par2.plot(t, i_m, color='green', label="I_m")
    p4, = par3.plot(t, v*1000, color='orange', label="V")

    lns = [p1, p2, p3, p4]
    host.legend(handles=lns, loc='best')

    # right, left, top, bottom
    par2.spines['right'].set_position(('outward', 50))
    par3.spines['right'].set_position(('outward', 100))
    # no x-ticks
    par2.xaxis.set_ticks([])
    par3.xaxis.set_ticks([])
    # Sometimes handy, same for xaxis
    #par2.yaxis.set_ticks_position('right')

    host.yaxis.label.set_color(p1.get_color())
    par1.yaxis.label.set_color(p2.get_color())
    par2.yaxis.label.set_color(p3.get_color())
    par3.yaxis.label.set_color(p4.get_color())

    plt.savefig("pyplot_multiple_y-axis.png", bbox_inches='tight')
```

In [41]:
```
params['t_spike'] = 0.5
params['dt'] = 0.0001
params['t_max'] = 1

# inhibitory
params['e_syn'] = -100e-3

v_inh, i_syn_inh, g_syn_inh, t = membrane_euler(mem_x, g_syn, passive_membrane_potential, i_syn_func, params, initial_
values)
print(max(g_syn_inh)*(10**9))
i_m_inh = i_mem_func(v_inh, params)
plot_func(g_syn_inh, i_syn_inh, i_m_inh, v_inh, t)



# excitatory
params['e_syn'] = 0
v_exc, i_syn_exc, g_syn_exc, t = membrane_euler(mem_x, g_syn, passive_membrane_potential, i_syn_func, params, initial_
values)
i_m_exc = i_mem_func(v_exc, params)
plot_func(g_syn_exc, i_syn_exc, i_m_exc, v_exc, t)
```

```
11.091889129491808
```





In [ ]: