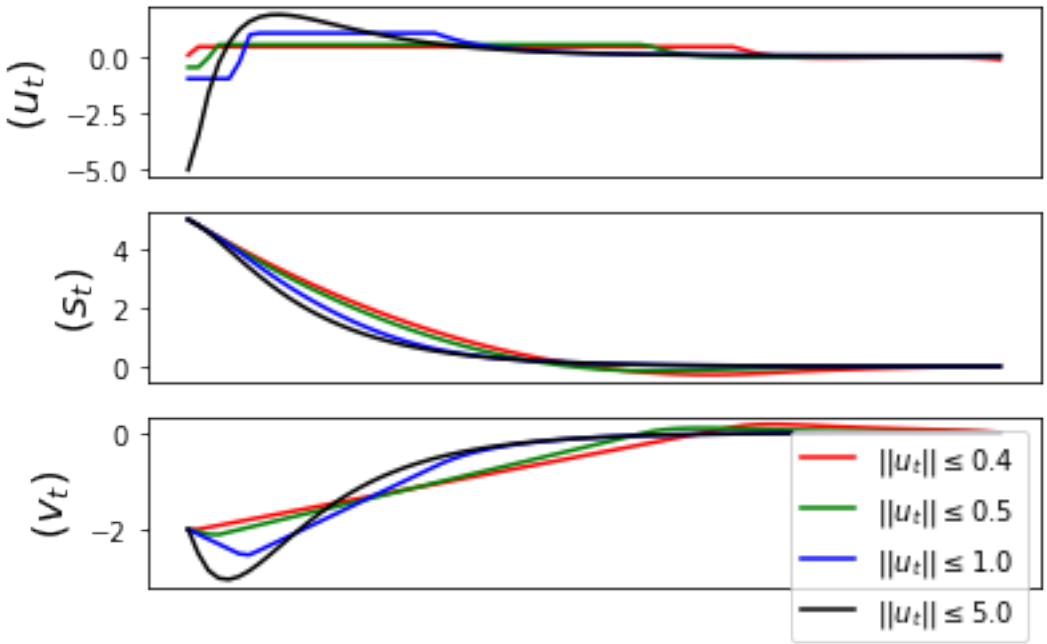
# Convex_Control_LQR_Box

August 25, 2022

[1]:
```
"""
        LQR with box constraints

"""

# Generate data for control problem.
import numpy as np

# state dimension of x_t
n = 2

# control dimension of u_t
m = 1

# time horizon T
T = 80

# time interval
delta_t = 0.1

# mass
m = 1

# dynamics matrix
A = np.array([[1, delta_t], [0, 1]])

# control matrix
B = np.array([[0],  [delta_t/m]])

# initial condition
x_0 = [5, -2]

# Form and solve control problem.
import cvxpy as cp

# state decision variable
x = cp.Variable((n, T + 1))
```

```
# control decision variable
u = cp.Variable((m, T))

traj_cost_scalar = 10.0
control_scalar = 1.0


# Plot results.
import matplotlib.pyplot as plt

f = plt.figure()
ax = f.add_subplot(311)

norm_u_max_list = [0.4, 0.5, 1.0, 5.0]
color_list = ['r', 'g', 'b', 'black']

for norm_u, col in zip(norm_u_max_list, color_list):

    # init cost
    cost = 0

    # init empty list of constraints and keep adding for DYNAMICS
    constr = []

    for t in range(T):
        # quadratic cost for the stage t
        cost += traj_cost_scalar * cp.sum_squares(x[:, t + 1]) + control_scalar␣
↪* cp.sum_squares(u[:, t])

        # dynamics constraints AND infinity norm constraint on control
        constr += [x[:, t + 1] == A @ x[:, t] + B @ u[:, t], cp.norm(u[:, t],␣
↪"inf") <= norm_u]

    # sums problem objectives and concatenates constraints.
    constr += [x[:, T] == 0, x[:, 0] == x_0]

    # solve LQR with constraints
    problem = cp.Problem(cp.Minimize(cost), constr)
    problem.solve()

    u_str = r'$\vert\vert u_t \vert\vert \leq ' + str(norm_u) + '$'

    # Plot (u_t)_1.
    plt.subplot(3,1,1)
    plt.plot(u[0, :].value, color=col, label=u_str)
    plt.ylabel(r"$(u_t)$", fontsize=16)
    plt.xticks([])
```

```
    # Plot (x_t)_1.
    plt.subplot(3,1,2)
    x1 = x[0, :].value
    plt.plot(x1, color=col, label=u_str)
    plt.ylabel(r"$(s_t)$", fontsize=16)
    plt.xticks([])


    # Plot (x_t)_2.
    plt.subplot(3,1,3)
    x1 = x[1, :].value
    plt.plot(x1, color=col, label=u_str)
    plt.ylabel(r"$(v_t)$", fontsize=16)
    plt.xticks([])

plt.legend()
plt.show()
```




[ ]: