Right now the pendulum cost is written in terms of residual away from 0

xPendCost = CostModelDoublePendulum(state, crocoddyl.ActivationModelWeightedQuad(np.array([1.] \* 4 + [0.1] \* 2)), nu)

Both the terminal cost and running cost is defined in terms of this

This creates a model of the cost of the Double Pendulum, given:

1. A model of its state
2. Weighted Quadratic Activation (Activation is a function that scales its inputs by some amount
3. Number of Controls

The cost model estimates the cost of being in a given state and inputting given controls at each point in time:

Cost(t) = activation(state(t), control(t))

The activation functions generally used are quadratic, and here we see we have a quadratic model, that is weighted:

* two joint angles and angular velocities punished by factor of 1
* controls punished by factor of .1