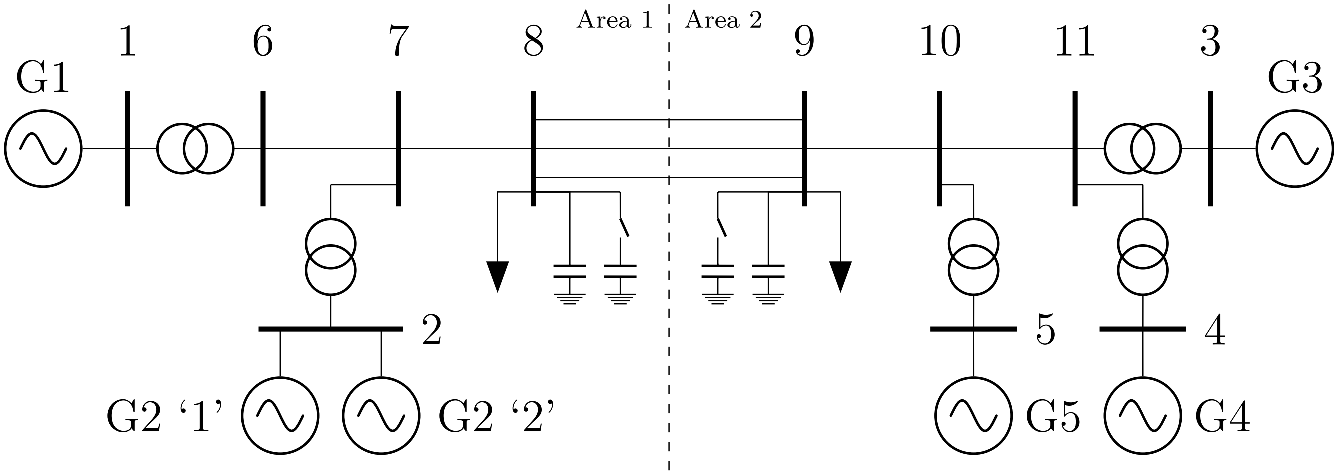
Test System:



Event Description: Load step of 75 MW on bus 9 when t=5.
Area 1 is scheduled to send 100 MW to Area 2 for the entire simulation.

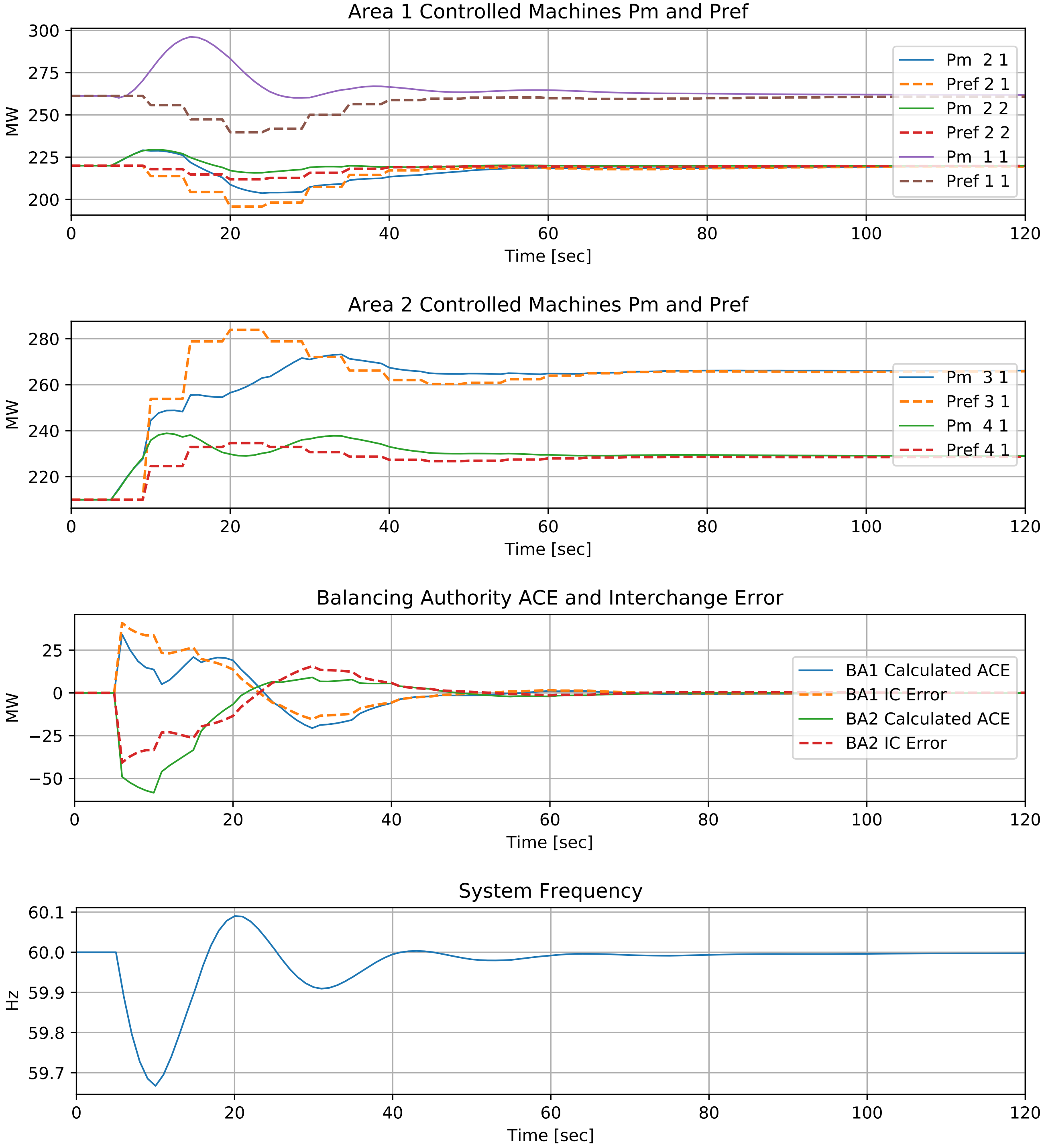
Simulation Agents: This simulation uses step, power plant, and balancing authority agents. An example of how such agents are defined in code is shown below.

```
# Perturbances
# Defined as a list of single quote strings.
mirror.sysPerturbances = [
    'load 9 : step P 5 75 rel', # Step Load P +75 MW relative at t=5
    ]

# Power Plants
# Defined as a dictionary of lists of strings
mirror.sysPowerPlants ={'pp1': ["gen 2 1: 0.75 : rampA", "gen 2 2 : 0.25: rampA"],
                        'pp2': ["gen 3 : 0.75: rampA", "gen 4 : 0.25: rampA"],
                        }

# Testing of Balancing Authority input
# Defined as a dictionary of dictionaries
mirror.sysBA = {
    'BA1':{
        'Area':1,
        'B':" 1.0 : p", # MW/0.1 Hz -> 1.0 percent of initial Area load
        'ActionTime': 5.00, # sends update signals as requred every x seconds
        'Type':'TLB : 0', # Tie-Line Bias : conditional ACE parts
        'Filtering': 'PI : 0.1 0.0001', # where 0.1 = Kp, and 0.0001 = a = Ki/Kp
        'CtrlGens': ['plant pp1 : .60 ',
                    'gen 1 : .40 : rampA']
        },
    'BA2':{
        'Area':2,
        'B':" 1.0 : p", # MW/0.1 Hz
        'ActionTime': 5.00,
        'Type':'TLB : 0', # Tie-Line Bias
        'Filtering': 'PI : 0.1 0.0001',
        'CtrlGens': ['plant pp2 : 1.0 ']
        },
    }
```

ACE Definition: Positive ACE denotes over generation. B (the frequency bias) is negative.

$$\begin{aligned} \text{ACE}_{\text{tie line}} &= P_{gen} - P_{load} - P_{\text{sched interchange}} \\ \text{ACE}_{\text{frequency bias}} &= 10B(f_{\text{actual}} - f_{\text{sched}})f_{base} \\ \text{ACE} &= \text{ACE}_{\text{tieline}} - \text{ACE}_{\text{frequency bias}} \end{aligned}$$

Initial Step ACE Results: Initial tests where calculated ACE was distributed as steps resulted with an undesirable and unrealistic system response.


Area 1 Controlled Machines Pm and Pref

Area 2 Controlled Machines Pm and Pref

Balancing Authority ACE and Interchange Error

System Frequency

Figure 1: Relevant Balancing Authority Simulation Data from Initial Test.

Ramp PI ACE Results: Changing the ACE distribution to ramps and using a PI controller to filter ACE with a proportional gain of 0.1 and integral gain of 10E-6 produces the responses shown in Figure 2. The filtered ACE is distributed to controlled machines according to their participation factor. Note that the 20 minute simulation uses a 1 second time step.

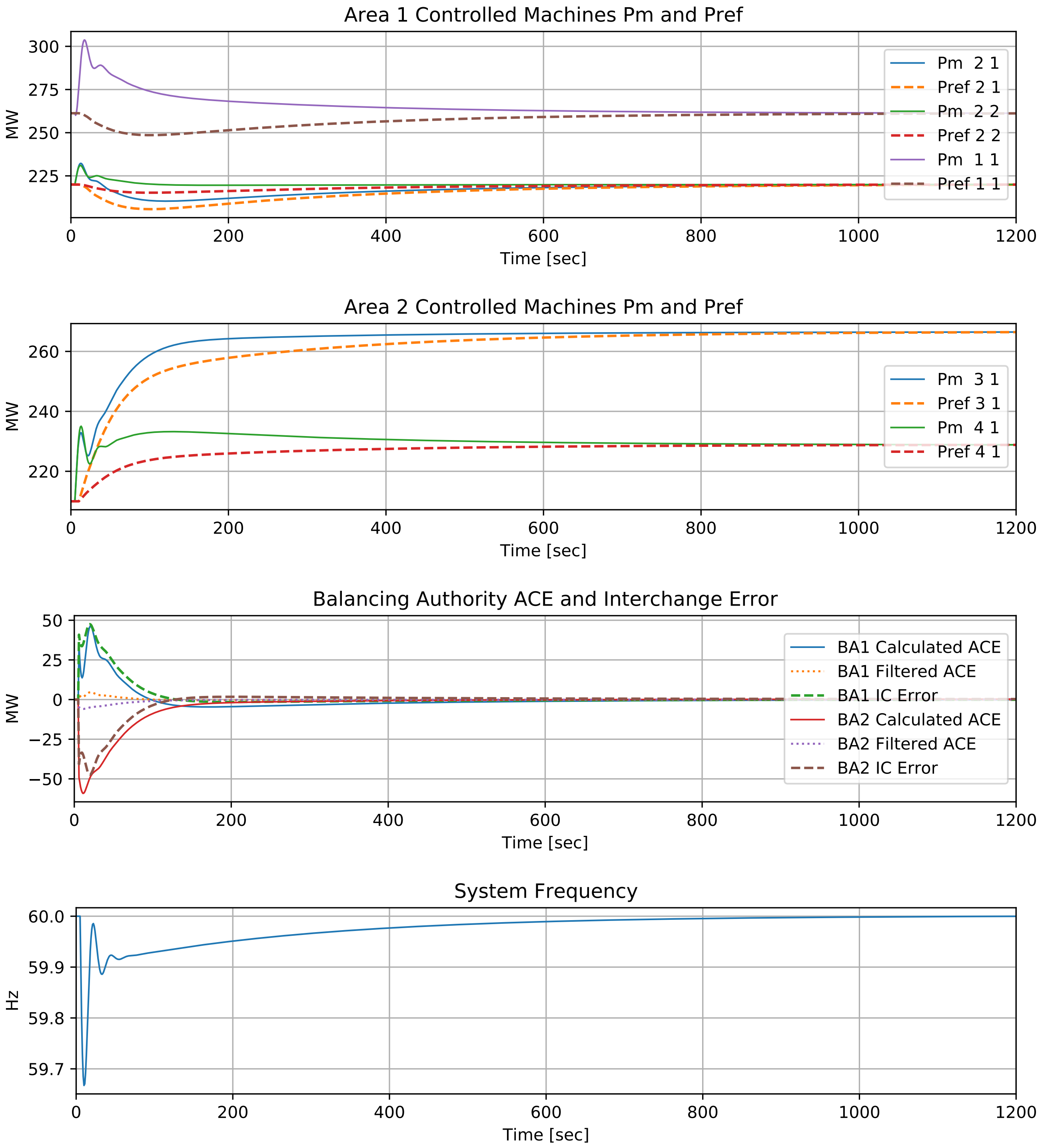


Figure 2: Relevant Balancing Authority Simulation Data Using Ramps and PI Filtering.

ACE Filtering Results 1: Using the previous PI controller, logic was added to always act on frequency bias ACE and only dispatch tie line ACE if it is the same sign as frequency deviation $(f_{actual} - f_{sched})$. This causes the system frequency to return to 60 Hz much faster, but also introduces minor frequency oscillations around t=180.

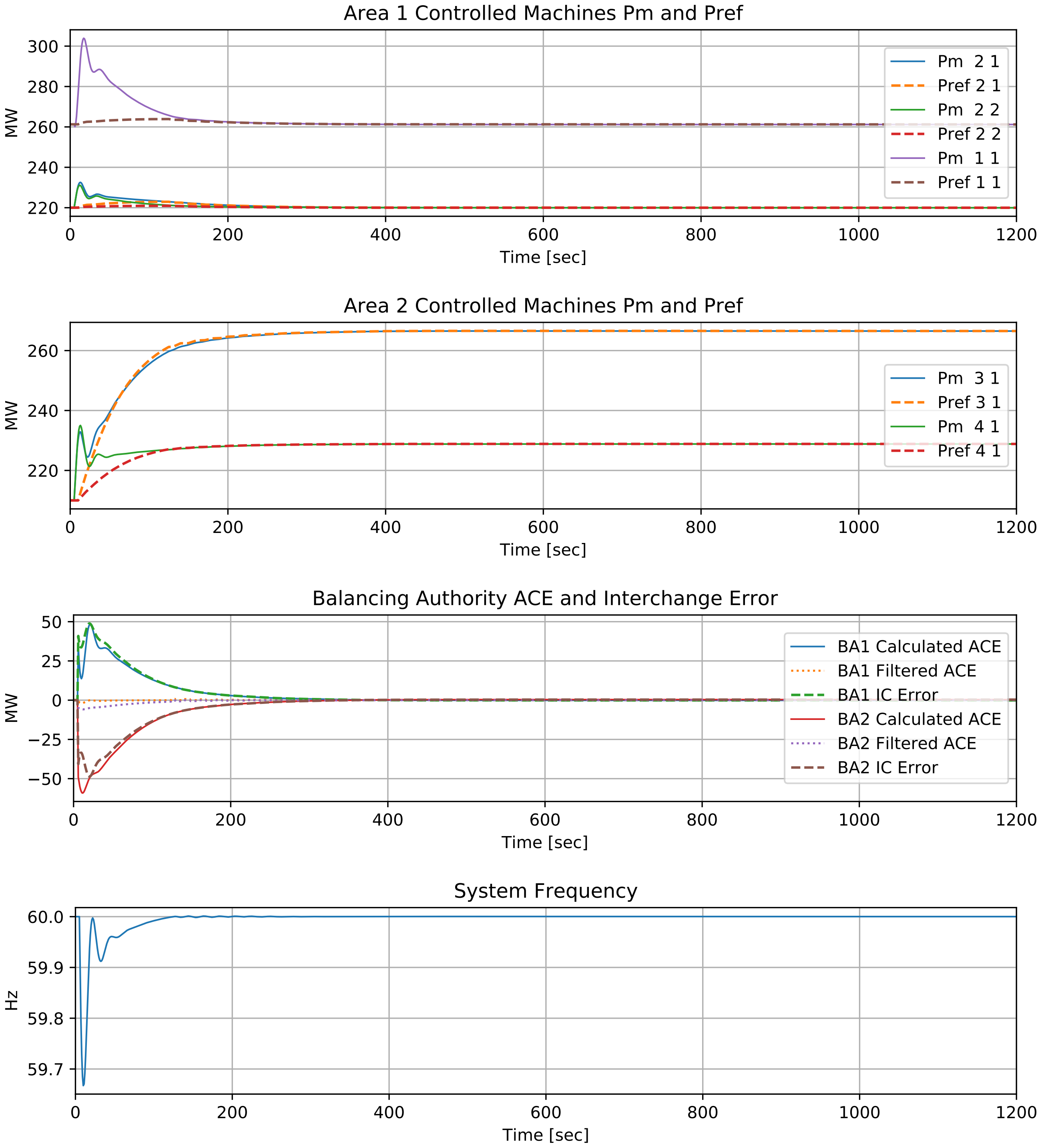


Figure 3: Relevant Balancing Authority Simulation Data Using Conditional Distribution.

ACE Filtering Results 2: Expanding on the previous filtering rules, ACE is only distributed when it is the same sign as frequency deviation. In this case, system recovery speed is still improved, frequency oscillations are not present near the steady state, and unnecessary Pref adjustment in Area 1 is eliminated.

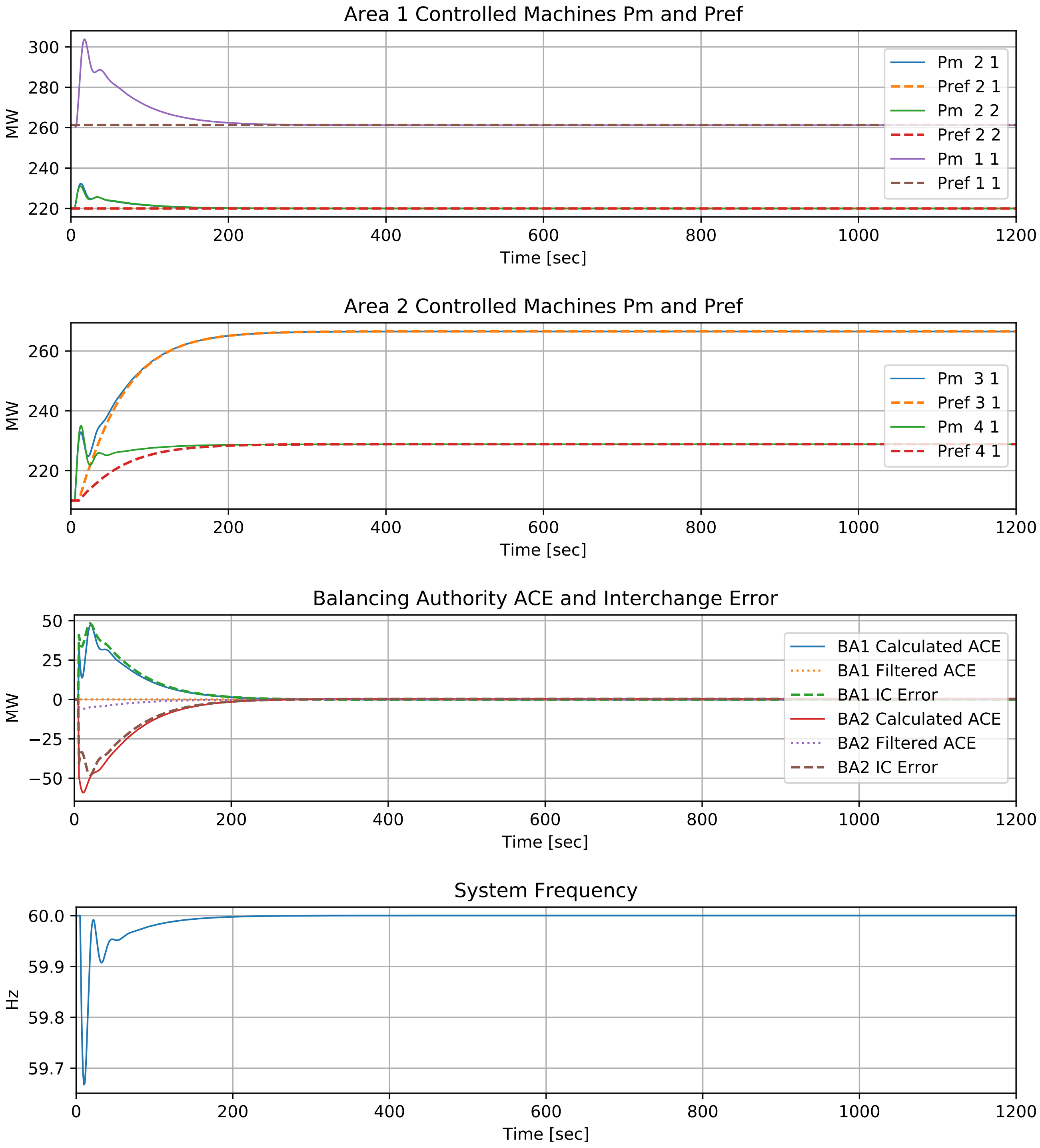


Figure 4: Relevant Balancing Authority Simulation Data With Refined Conditional Distribution.

ACE Filtering Results 3: Again expanding on the previous filtering rules, proportional controller gain was reduced to 0.02 while the integral gain was set at 2E-6 (80% reduction in both cases). This allows for a clearer definition between governor and AGC action without the use of deadbands.

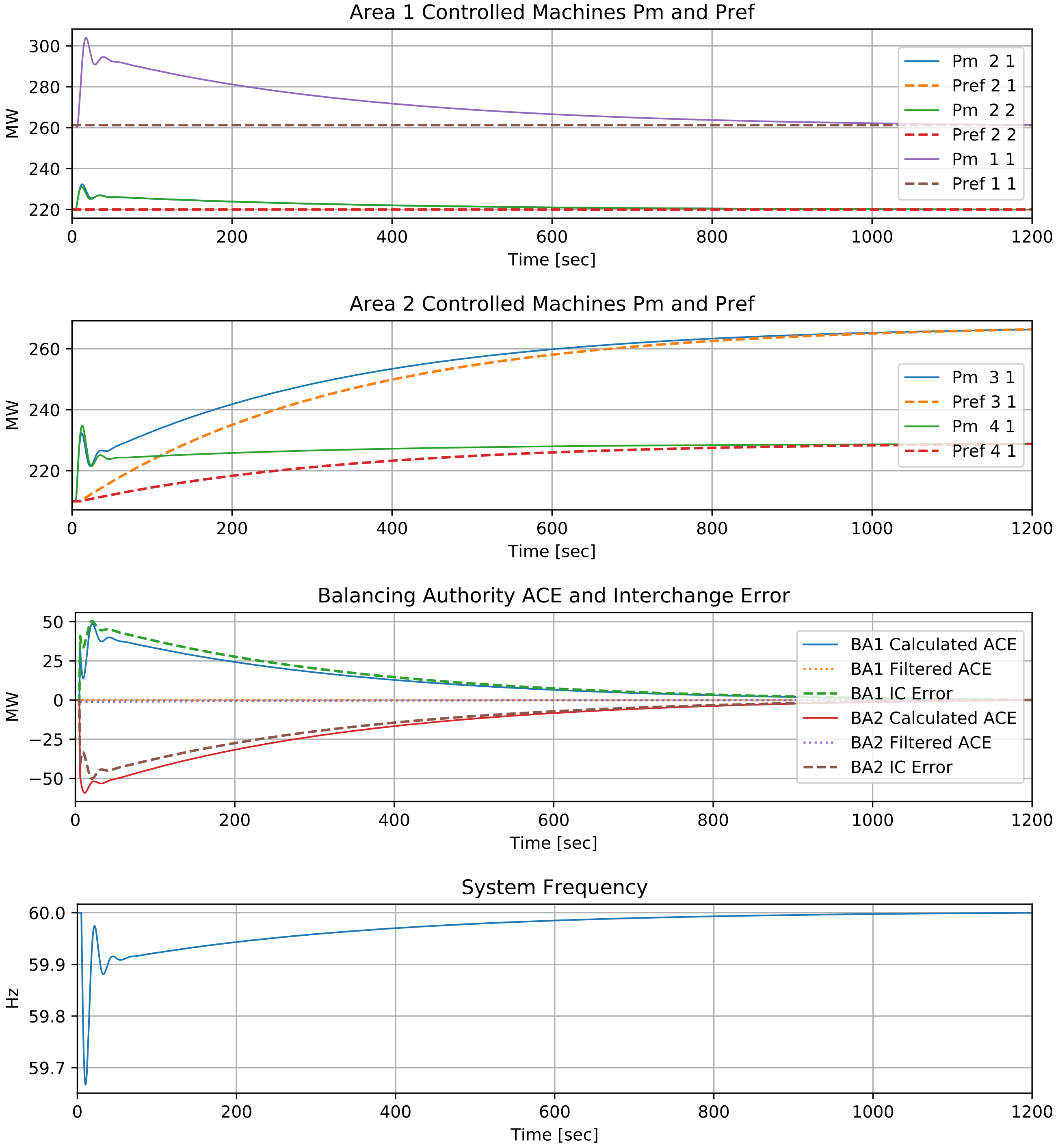


Figure 5: Relevant Balancing Authority Simulation Data with Conditional Distribution and Reduced Control Effort.

Quasi-Conclusions: There are many ways to manipulate distribution of ACE to achieve a desired system response. Simulations are a good way to verify system operation.

PSLTDSim (the software I've been writing for the past 7-ish months) is a simulation tool that can perform various long-term control experiments involving governors and AGC.