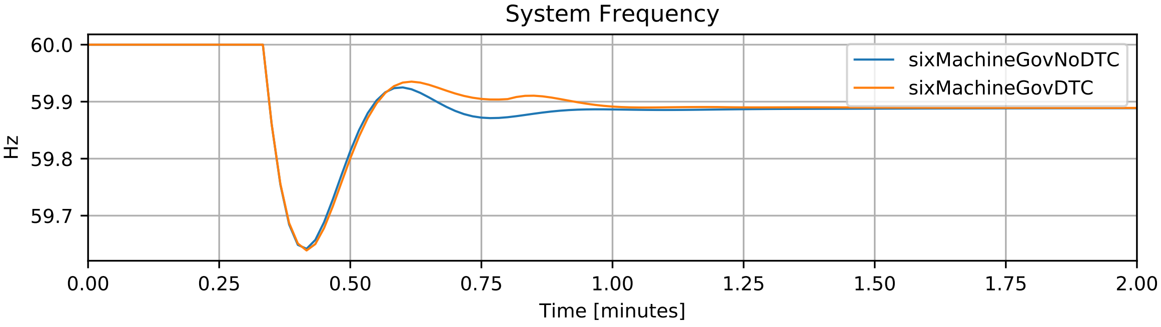
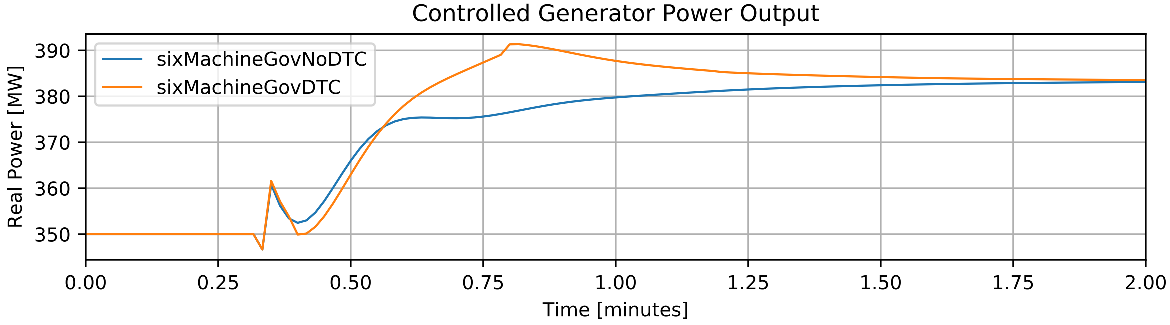
Initial Definite Time Controller (DTC) BPA Result Summary

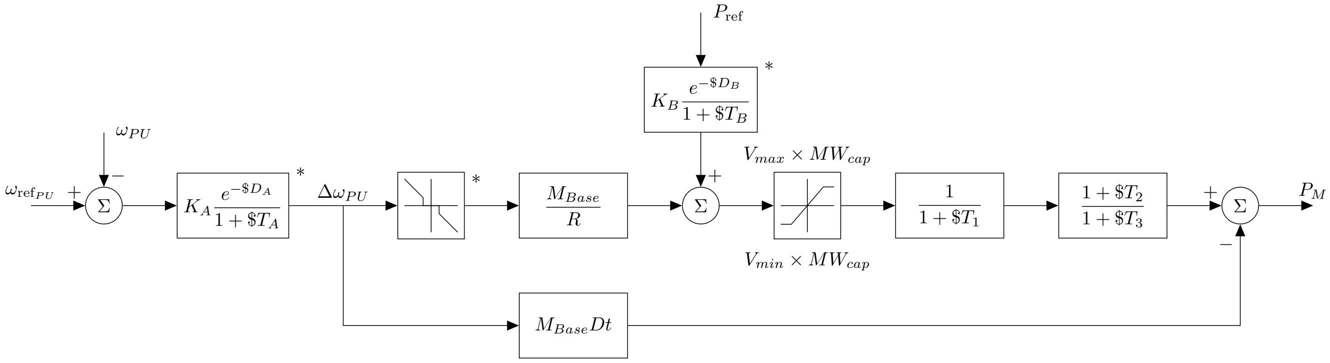
Using a six machine system, results show that the method of stepping Pref while also gaining input $\Delta\omega_{PU}$ produces similar generator output action provided by BPA as an 'undesirable' response. Remaining differences believed to be caused by differences in model size, time constants, and actual 'feed-forward' action.





Modified Governor Model

Input ω block moved so that gain would be more logical.


P_{ref}
$K_B \frac{e^{-\$D_B}}{1+\$T_B}$
*
ω_{PU}
−
$\omega_{\text{ref}_{PU}}$
+
Σ
$K_A \frac{e^{-\$D_A}}{1+\$T_A}$
*
$\Delta\omega_{PU}$
*
$\frac{M_{Base}}{R}$
+
Σ
$V_{max} \times MW_{cap}$
$V_{min} \times MW_{cap}$
$\frac{1}{1+\$T_1}$
$\frac{1+\$T_2}{1+\$T_3}$
+
−
Σ
P_M
$M_{Base}Dt$

Code Example and Explanation Code used to define system step, governor delay, and DTC action is provided below. In practice, this code is user defined in the simulation .ltd.py file.

An ungoverned generator on bus 5 has mechanical power stepped down 100 MW at t=20 to simulate the tripping of a generator. A governor delay block is used to gain the ω input by 0.5. DTC action occurs every 24 seconds (so that first action is near frequency nadir) and sets $P_{ref} = P_{ref0} + \dfrac{\Delta\omega}{R} M_{base} * 0.5$.

```
# Perturbances
mirror.sysPerturbances = [
    'gen 5 : step Pm 20 -100 rel', # Step no-gov generator down
    ]

# Delay block used as delta_w gain
mirror.govDelay ={
    'delaygen2' : {
        'genBus' : 2,
        'genId' : '1', # optional
        'wDelay' : (0, 0, .5),
        'PrefDelay' : (0, 0)
        },
    #end of defined governor delays
    }

# Definite Time Controller Definitions
mirror.DTCdict = {
    'bpaTest' : {
        'RefAgents' : {
            'ra1' : 'mirror : f',
            'ra2' : 'gen 2 1 : R',
            'ra3' : 'gen 2 1 : Pref0',
            'ra4' : 'gen 2 1 : Mbase',
            },# end Referenc Agents
        'TarAgents' : {
            'tar1' : 'gen 2 1 : Pref',
            }, # end Target Agents
        'Timers' : {
            'set' :{ # set Pref
                'logic' : "(ra1 > 0)", # should always eval as true
                'actTime' : 24, # seconds of true logic before act
                'act' : "tar1 = ra3 + (1-ra1)/(ra2) * ra4 *0.5 ", # step Pref
            },# end set
            'reset' :{ # not used in example
                'logic' : "0",
                'actTime' : 0, # seconds of true logic before act
                'act' : "0", # set any target On target = 0
            },# end reset
            'hold' : 0, # minimum time between actions (not used in example)
            }, # end timers
        },# end bpaTest
    }# end DTCdict
```