Document Purpose / Intent

An extended term case may require the insertion of new, or additional, modeled generation resources. There is currently no known method in PST to increase inertial generation during a simulation. Additionally, there seemed to be minor confusion/forgetfulness as to how PST currently handles tripping a generator. A closer look into what happens when machines are tripped off may provide some insight into a method to 'un-trip', or insert, generators.

How generators are currently tripped in PST

During simulation initialization, `g.mac.mac_trip_flags` is initialized as a row vector of zeros that correspond to the `mac_con` array, and `g.mac.mac_trip_states` is set to zero. To trip a generator, a `mac_trip_flag` is set to 1 via the user generated `mac_trip_logic` code. The `mac_trip_logic` is executed in the `initStep` function which alters `g.mac.mac_trip_flags` to account for any programmed trip. Specifically, a 0 in the `g.mac.mac_trip_flags` row vector is changed to a 1 to signify a generator has tripped.

The `g.mac.mac_trip_flags` vector is summed in the `networkSolution` (`networkSoltuionVTS`). If the resulting sum is larger than 0.5, the line number connected to the generator in the `g.line.line_sim` is found and the reactance is set to infinity (1e7). The reduced y matrices are then recalculated and used to solve the network solution via an `i_simu` call.

If derivatives of the tripped machine are not set to zero (as the current VTS method does), the generator's speed increases, mechanical power output eventually drops to zero/near-zero, and the attached exciter Efd appears to approach 1.

Realistically, Pmech and all P and Q limits should also probably be set to zero which may 'clean up' values.

It should be noted that if a machine is tripped, new reduced Y matrices are created every simulation step. This repeated action could be reduced via use of globals.

- create new globals for something along the lines of 'Y sim' variables that are purposefully selected each network solution.
- create an `old_mac_trip_flags`
- compare `old_mac_trip_flags` to `mac_trip_flags`
- if there has been a new trip, update associated Y matrix globals
- if there hasn't been a new trip, but a machine has tripped, use the stored Y matrix

This is merely a calculation reduction and not required.

Initial 'Un-trip' Thoughts

If the trip flag is changed from 1 to zero - the reactance would return to normal, essentially reconnecting the generator to the system. However, the calculated states and derivatives would be no longer be synchronized and probably cause all sorts of issues/transients.

The machine model may be re-initialized via the use of the 0 flag and setting i to the correct number of the generator to re-initialize. This would require knowing what type of machine model is tripped so the proper function call is used. The effects of such action is untested - a simple test case should be created and attempted.

Basically: create small system with multiple generators, trip a generator, allow system to regain steady state, reconnect generator by setting machine trip flag back to zero. Observe and attempt to fix issues.

'Un-trip' Method 1 - Alter Aggregate Generator

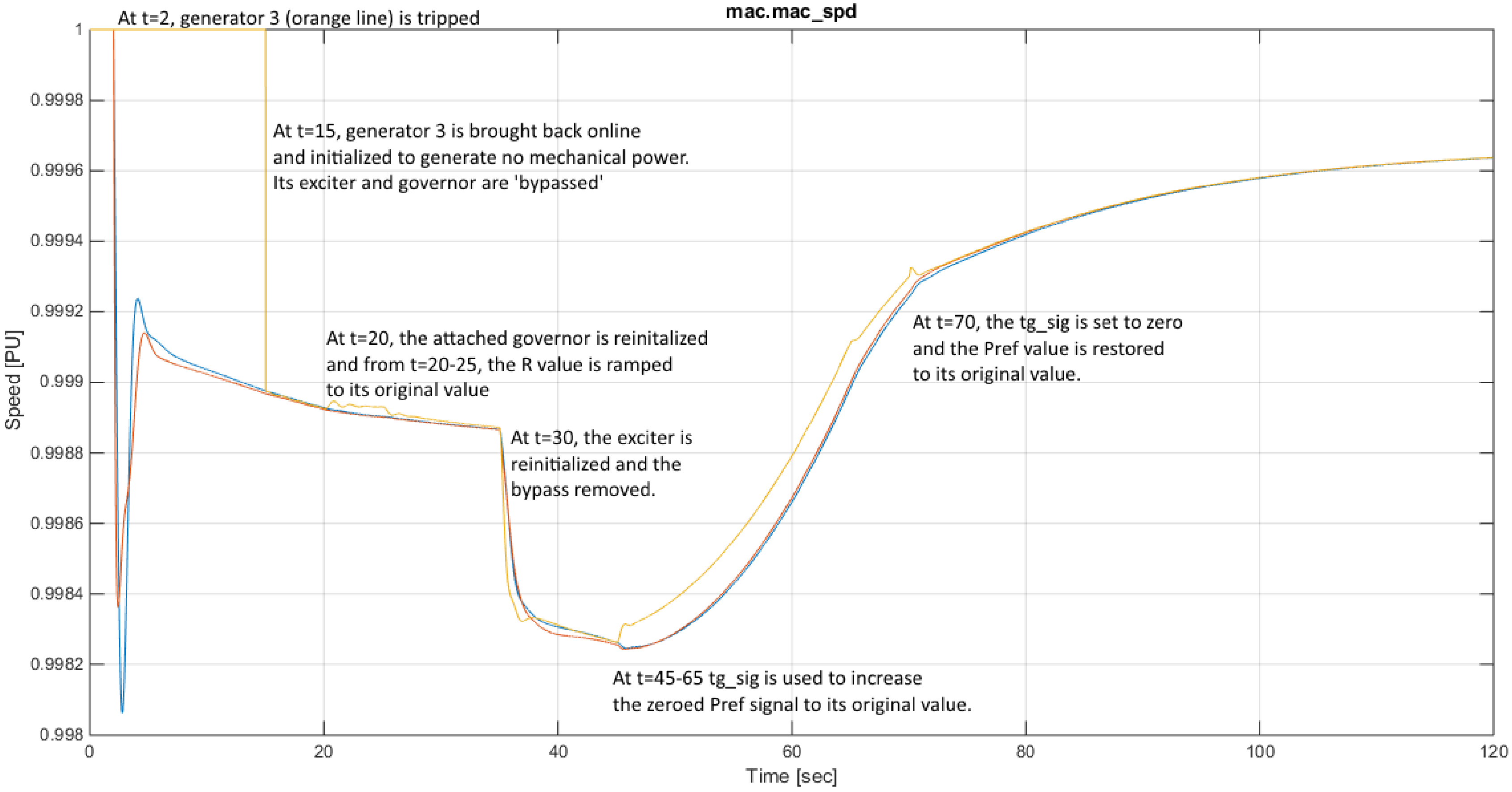
This method includes altering a machine's base during simulation to imitate the addition/reduction of an aggregate generator. It would require recalculation of all `mac_pot` values used in model calculations. Additionally, the monitoring of what is actually occurring may be obfuscated by the fact that any gains in capacity would be added to the existing aggregate value. This method was not predicted as being viable due to foreseen complexity/unavoidable transients.

'Un-trip' Method 2 - Stand alone Generator

Account for all generators that may, or may not, be added into the system during creation of the system data file (case). Start with Pref = Pmech = 0, turn exciter 'off', and ≈ 0 inertia. While inertia is essentially 0, generator will likely respond very fast to any input. Inertia step and Pref altered. Essentially initialize a generator with zero Pmech to the connected bus with speed the same as other generators in the area/system...

To track capacity/ineria: use -1 flag to denote a generator that is tripping from 1 to 0.

Experimental Results Using the previous notes as a guide, a 3 machine scenario was created where 1 generator will trip off, then attempt to be reconnected with minor transient behavior. Initial results are appear promising. All transient behavior may not be eliminated, but could possibly reduced by ramping signals (specifically the exciter).



At t=2, generator 3 (orange line) is tripped

At t=15, generator 3 is brought back online and initialized to generate no mechanical power. Its exciter and governor are 'bypassed'

At t=20, the attached governor is reinitalized and from t=20-25, the R value is ramped to its original value

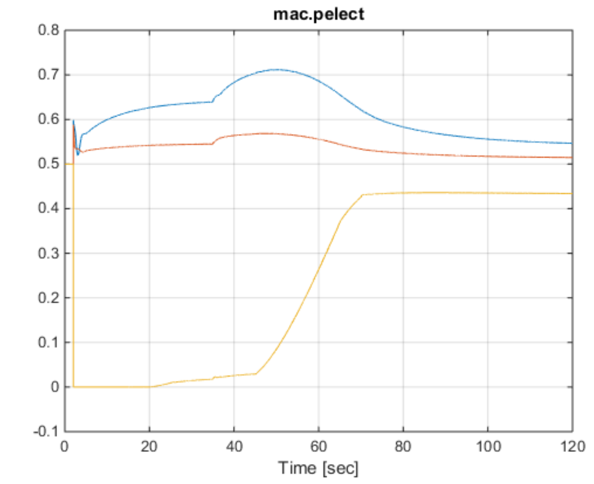
At t=30, the exciter is reinitialized and the bypass removed.

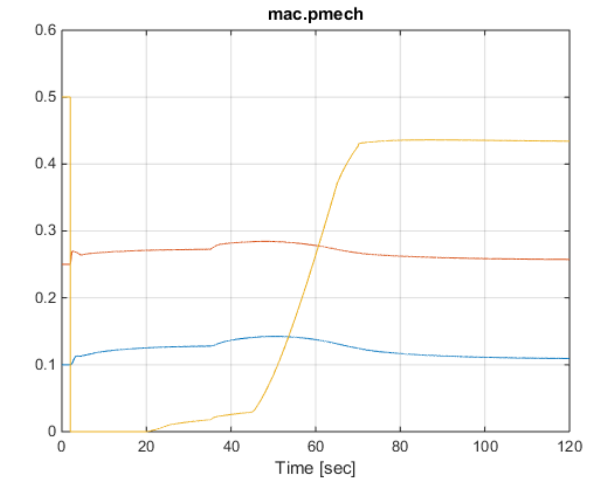
At t=45-65 tg_sig is used to increase the zeroed Pref signal to its original value.

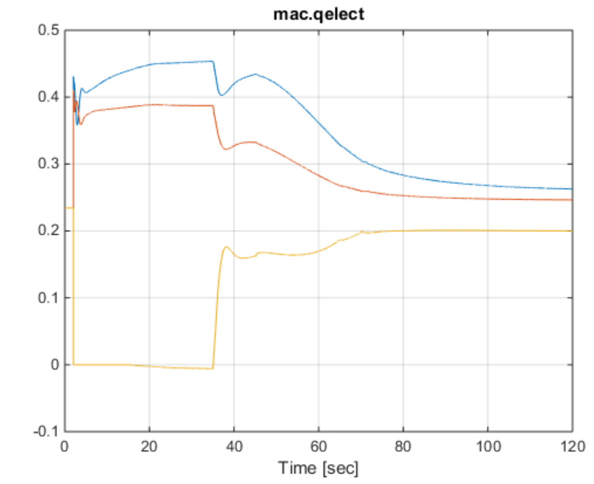
At t=70, the tg_sig is set to zero and the Pref value is restored to its original value.

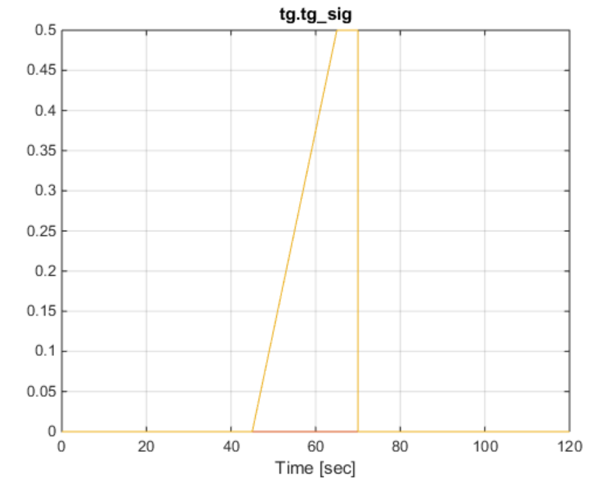
The exciter ramping up field voltage(and generating vars) caused the speed dip at 35. The exciter is required to be connected before mechanical power is ramped, else voltage collapse occurs and the machine ‘runs away’. The machine speeds do not return to 1 as the mechanical powers do not fully restore. The P_{ref} setting of the tripped governor appears to not end at the 0.5 it is set to, or the value is scaled somewhere in the model that has been missed.

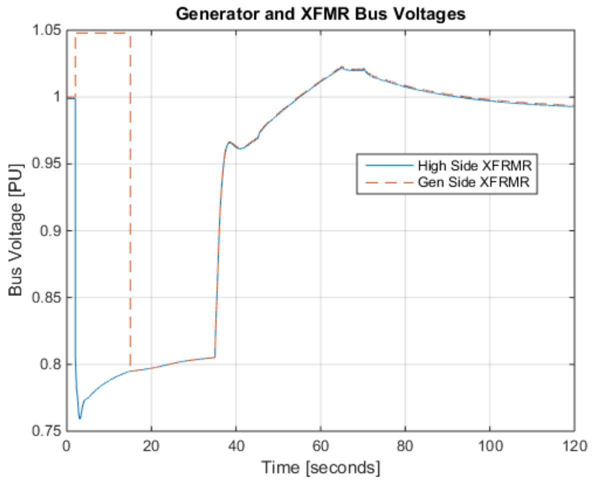
Results Continued

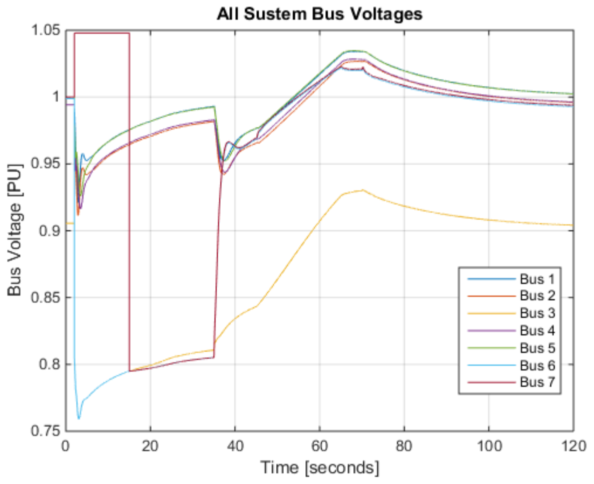

mac.pelect
Time [sec]


mac.pmech
Time [sec]


mac.qelect
Time [sec]


tg.tg_sig
Time [sec]


Generator and XFMR Bus Voltages
Bus Voltage [PU]
Time [seconds]
High Side XFRMR
Gen Side XFRMR


All Sustem Bus Voltages
Bus Voltage [PU]
Time [seconds]
Bus 1
Bus 2
Bus 3
Bus 4
Bus 5
Bus 6
Bus 7