So I ran some tests on various FMU method. Below is my summary.

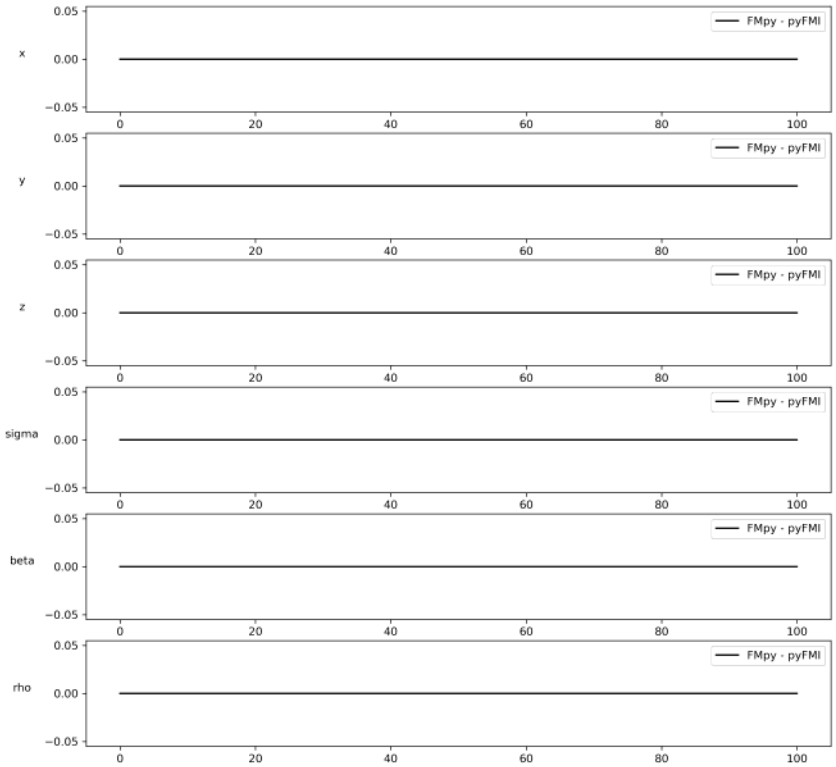
Conclusion:

1. Simulations performed via FMU outside of Dymola produce the same results as simulations performed from the original code within Dymola. However, you must be very careful that what you are simulating in Dymola is exactly the same as the FMU.
2. The way we are running FMUs in Dymola is wrong.
3. No effort was made in this study to compare the simulation performance between methods (i.e., simulation time, events, modeling control, etc.)

Supporting Explanation:

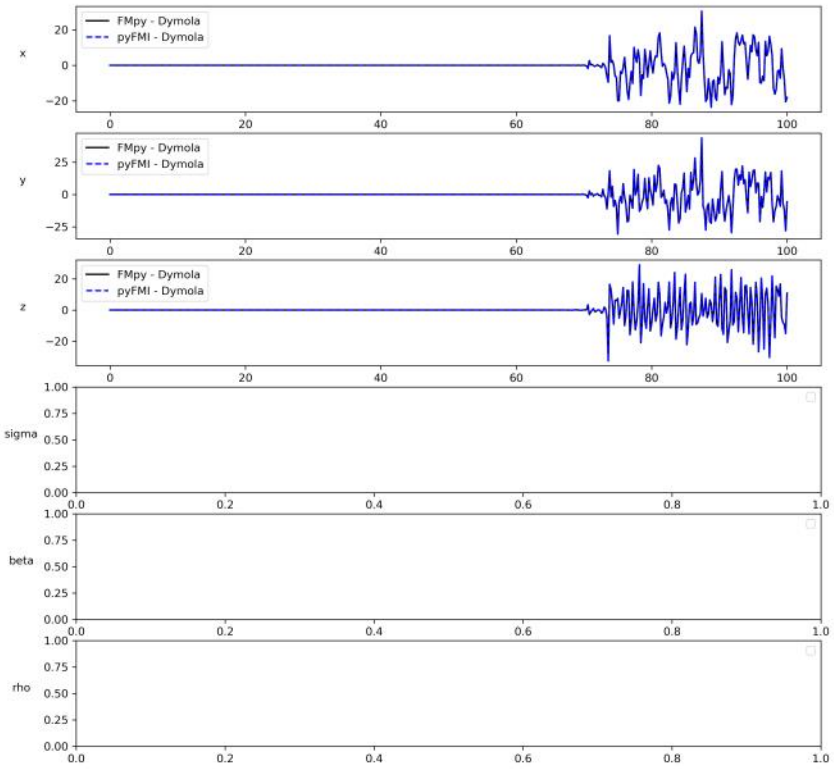
I ran several simulations to find where and if FMUs differed from Dymola. Overall this was a single mathematical problem (Lorenz System) codified based on their parameter/input format only into 3 Modelica models and 3 FMUs created using the Modelica models. These models were then simulated in various scenarios to determine when and if FMU results differentiated from Dymola source code model results.

1. The simulations I ran were using FMpy and pyFMI to independently run FMUs from python. These always produced the same result, for all cases described below. That is good. Below is an example output of that comparison.

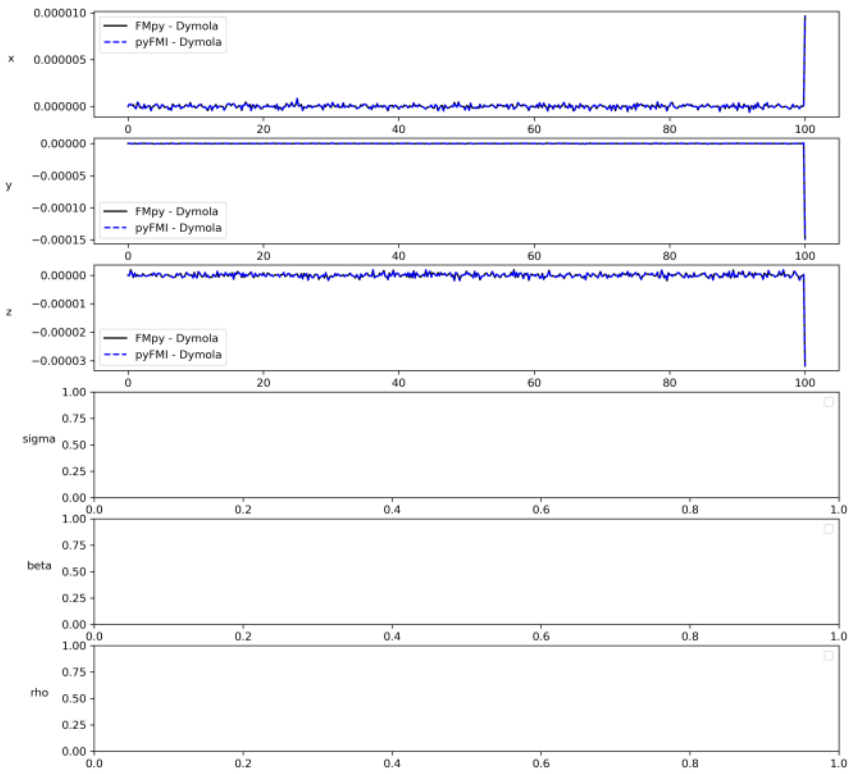


2. I compared FMpy and pyFMI to Dymola produced results (non-fmu in Dymola - i.e., original code). I did this two ways:
    a. Compared a single simulated FMU to the result from a dymola model containing each of the

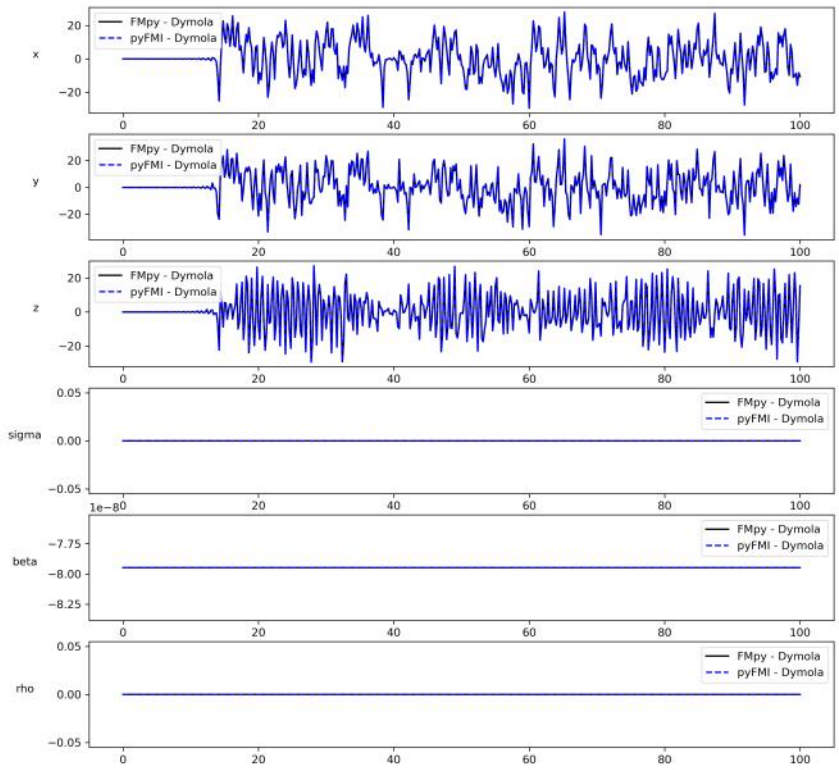
3 variations of models. See BasicTest.mo as an example. This produced results that were identical for a while and then diverged at some later point in the simulation (see figure). No attempt was made to identify if this could be corrected



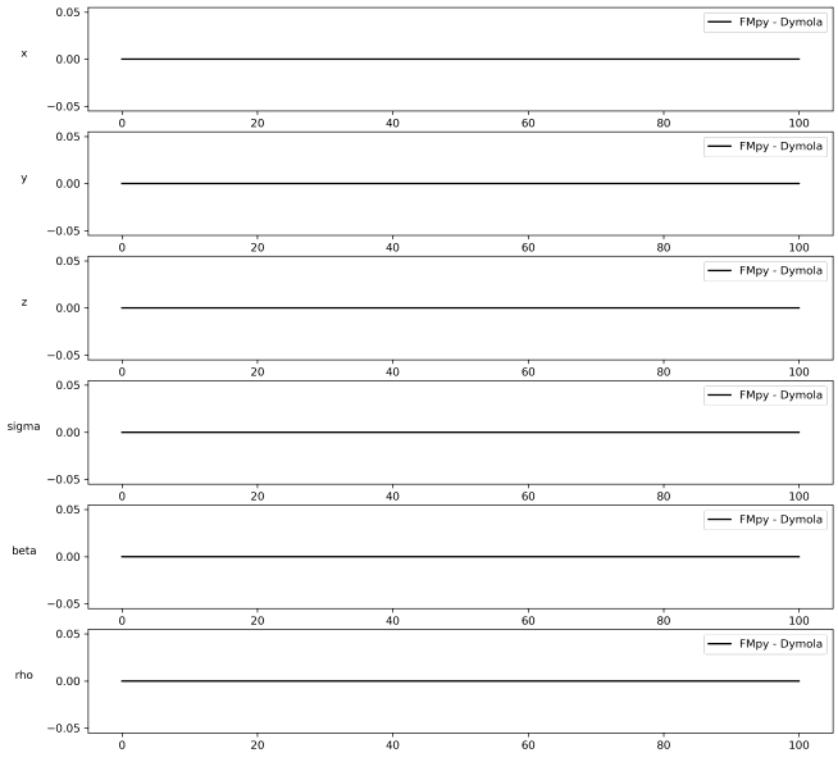
b. Compared a single simulated FMU to the result from a dymola model containing only the single model of interest. See Parameters.mo as an example. This produced identical results throughout the simulation with minor error at the last simulation point. Not sure why this occurs but perhaps could be easily accounted for by throwing away the last value, or trying to figure out what is really going on there. In any case, this is GOOD NEWS!



3. I compared FMpy and pyFMI to Dymola produced FMU results (fmu in Dymola - i.e., FMU placed in model and simulated). I did this two ways:
    a. Compared a single simulated FMU to the result from a dymola model containing each of the 3 variations of FMUs. See FMU_BasicTest.mo as an example. This produced results that were identical for a while, though for much less time than 2.a (above) and then diverged at some later point in the simulation (see figure). No attempt was made to identify if this could be corrected.



b. Compared a single simulated FMU to the result from a dymola model containing only the single equivalent FMU of interest. See FMU_Parameters.mo as an example. This produced erroneous results that appear similar to 3.a (see figure below… appears the are performing the same). This leads me to believe that the current method employed in simulating FMUs is doing something different that we intend.



i. For clarity, the current method is to import an FMU. Drag and drop it into a model. Using the simulation setup as for a normal model, set the simulation settings and simulate.
    1) I think this does not work because it is wrapping the FMU (which is in co-simulation mode) in another solver, thereby causing errors. Tried running a FMU_Parameters directly without putting it in a model… it still produced the same errors.