So you want to linearize? - Version 3

I. Before Linearization:

For the operating point in which you are interested, get steady state values for all initial conditions you may be interested in (the initial conditions of interest will be in ElastoDyn.dat).
The initial conditions that need to be updated are:

- Blade Pitch BlPitch
- Rotor Speed RotSpeed
- Platform Roll,Pitch,Yaw,Surge,Sway,Heave Ptfm...

Update the initial conditions in ElastoDyn to be as accurate as possible; doing so will make your linearization run faster as well as help to avoid solver problems in FAST during its initial transients. With the most accurate initial conditions, determine how long until your signals reach a steady state. The linearization process runs OpenFAST so you want to make sure that all initial transients have ended and the system is operating in steady-state for your operating point! So if even with your initial conditions updated, and it still takes 500 seconds for your simulation to reach steady state, then your starting linearization time should be 500 seconds.

II. To Linearize:

A. ServoDyn.dat

none PC_Mode = 0 (uses a fine pitch value of the initial blade pitch value set in ElastoDyn)
simple VS VS_Contrl = 1 (uses simple variable-speed torque control)

In ServoDyn, under the SIMPLE VARIABLE-SPEED TORQUE CONTROL section update the variables for the torque control, they may be initialized to 9999.9 for now, but an external controller (such as rosco or Simulink) is not used in the simulation, so this torque controller will be used for linearization (seen below). or set to values from rosco/simulink controller

Note: these are the values for the current USFLOWT 10MW turbine

```
--- SIMPLE VARIABLE-SPEED TORQUE CONTROL ---
0.0001                                  VS_RtGnSp
207234.9879000                          VS_RtTq
0.0001                                  VS_Rgn2K
0.0001                                  VS_SlPc
```

If using a rosco or simulink controller, you can find the value for the rated torque (VS_RtTq) in your controller parameters (the variable names should be similar for whatever type of controller you are using).

Note: When VS_RtGnSp, VS_Rgn2K,VS_SlPc are very small numbers, the torque will be held constant at the value set for VS_RtTq. If linearizing in Region 2, then VS_RtTq should be set to the steady state torque value at the operating point of interest.

B. ElastoDyn.dat

To simplify the verification process, we recommend starting with a simple 1 DOF linearization with only the GenDOF set to True. After this has been verified, additional DOF's can be enabled. The DOF's determine which states are included in the linearization output.
Update initial conditions to be steady state values at the operating point you are looking at.

C. HydroDyn.dat

WaveMod= 0 (still water)

i.e. no disturbances

D. InflowWind.dat

WindType = 1 (steady wind)

Change HWindSpeed to the wind speed corresponding to the operating point for your linearization.

E. Model.fst

Note: the line under NLinTimes is cut off, but there are 36 different times there.

```
--- LINEARIZATION ---
True              Linearize          L
False                CalcSteady
3                    TrimCase
0.1                TrimTol
0.01                 TrimGain
0                    Twr_Kdmp
0                    Bld_Kdmp
36                    NLinTimes
500 500.173520735728     500.34704147
1                    LinInputs
1                    LinOutputs
False                LinOutJac
False                LinOutMod
```

NLinTimes = 12
omega =

To get LinTimes find how long it takes for your rotor to turn a full rotation (use 10 degrees if you want to linearize at 36 rotor positions, 20 degrees for 18 rotor positions, etc), (based off the rotor speed at your selected operating point); we will call this delta_r. Suppose your starting time is t0 = 500. To get all the linearization times you will do t(i)=t0+i*delta_r, where i = {0, 1, ... NLinTimes-1}

NREL recommends linearizing at 36 azimuth positions separated by 10 deg, so you will have 36 values defined for LinTimes.

Make sure TMax is a few seconds longer than the maximum value in your list of LinTimes.

Run the simulation as you would normally (either through the command line or simulink). You just have to run it once, and you should have 36 linearization output files. (Model.1.lin, Model.2.lin, …, Model.36.lin)

Each of these linearization files contains the inputs, outputs, states, and A, B, C and D matrices corresponding to the specific azimuth position.

III. After Linearization:

Note: Since the linearization processes returns 36 state-space models, post-processing is required to find an azimuth-averaged linearized system. NREL provides the 'runMBC' tool for this post-processing.

A. Download matlab-toolbox from OpenFAST github
B. Use runmbc.m
C. Modify the paths and number of linearization files as necessary (see figure below)

```
%addpath(genpath('c:\Users\njohnso1\Documents\matlab-toolbox-Manu\matlab-toolbox\')); % update this path
addpath(genpath('C:\Users\eleny\code\ROSCO_toolbox-develop\Matlab_Toolbox'));
addpath(genpath('C:\Users\eleny\LINEARIZATION\OpenFAST-Dev2.8a_Models_Dist\13m-Cans\SubDyn-MAP\Hs1.13-WT11.4\20msLIN'));
%addpath(genpath('C:\Users\eleny\controller_verf'));
addpath(genpath('C:\Users\eleny\LINEARIZATION\matlab-toolbox-master'));
LinTimes = 36; % number of lin files you have
newFSTName = 'Model.fst'; % change this accordingly
%newFSTName = 'NREL5MW_DAC.fst';
FileNames = strcat( strrep(newFSTName, '.fst','.'), strrep( cellstr(num2str( (1:LinTimes)' )), ' ', ''), '.lin' );
fx_mbc3(FileNames)
```

The runMBC gives you the average across the NLinTimes azimuth positions.. It will have AAvg, BAvg, CAvg, DAvg, etc.

D. Selecting States

To select specific states (all but the Craig-Bampton modes and generator position) look at the DescStates cell of the runMBC output struct. The DescStates is the order of the states. So, to select the states you want, choose those indices that correspond to the states described in the DescStates cell array.

Note: the generator position state results in an unstable eigenvalue at zero for the AvgA matrix based on the torque-to-position relationship and that it is common practice to eliminate this state since generator position is not usually used in control.

Below is a figure of the output from runMBC.m. DescStates has the order of the states, because they are in a different order than what is in the linearization file.

```
              DescStates: {56×1 cell}
                   ndof2: 28
                   ndof1: 0
            RotSpeed_rpm: 9.5828
               WindSpeed: 20
performedTransformation: 0
                       A: [56×56×8 double]
                       B: [56×127×8 double]
                       C: [896×56×8 double]
                       D: [896×127×8 double]
                    AvgA: [56×56 double]
                  eigSol: [1×1 struct]
                    AvgB: [56×127 double]
                    AvgC: [896×56 double]
                    AvgD: [896×127 double]
```

Inside DescStates you will see this (this example has all the DoF turned on, so the maximum number of states). For this example you would choose the states (1:11, 12:22, 47:68).

ans.DescStates

	1
1	ED Platform horizontal surge translation DOF (internal DOF index = DOF_Sg), m
2	ED Platform horizontal sway translation DOF (internal DOF index = DOF_Sw), m
3	ED Platform vertical heave translation DOF (internal DOF index = DOF_Hv), m
4	ED Platform roll tilt rotation DOF (internal DOF index = DOF_R), rad
5	ED Platform pitch tilt rotation DOF (internal DOF index = DOF_P), rad
6	ED Platform yaw rotation DOF (internal DOF index = DOF_Y), rad
7	ED 1st tower fore-aft bending mode DOF (internal DOF index = DOF_TFA1), m
8	ED 1st tower side-to-side bending mode DOF (internal DOF index = DOF_TSS1), m
9	ED 2nd tower fore-aft bending mode DOF (internal DOF index = DOF_TFA2), m
10	ED 2nd tower side-to-side bending mode DOF (internal DOF index = DOF_TSS2), m
11	ED Nacelle yaw DOF (internal DOF index = DOF_Yaw), rad
12	ED Variable speed generator DOF (internal DOF index = DOF_GeAz), rad
13	ED Drivetrain rotational-flexibility DOF (internal DOF index = DOF_DrTr), rad
14	ED 1st flapwise bending-mode DOF of blade 1 (internal DOF index = DOF_BF(1,1)), m
15	ED 1st flapwise bending-mode DOF of blade 2 (internal DOF index = DOF_BF(2,1)), m
16	ED 1st flapwise bending-mode DOF of blade 3 (internal DOF index = DOF_BF(3,1)), m
17	ED 1st edgewise bending-mode DOF of blade 1 (internal DOF index = DOF_BE(1,1)), m
18	ED 1st edgewise bending-mode DOF of blade 2 (internal DOF index = DOF_BE(2,1)), m
19	ED 1st edgewise bending-mode DOF of blade 3 (internal DOF index = DOF_BE(3,1)), m
20	ED 2nd flapwise bending-mode DOF of blade 1 (internal DOF index = DOF_BF(1,2)), m
21	ED 2nd flapwise bending-mode DOF of blade 2 (internal DOF index = DOF_BF(2,2)), m
22	ED 2nd flapwise bending-mode DOF of blade 3 (internal DOF index = DOF_BF(3,2)), m
23	SD Craig-Bampton mode 1 amplitude, -
24	SD Craig-Bampton mode 2 amplitude, -

ans.DescStates

	1
46	SD Craig-Bampton mode 24 amplitude, -
47	ED First time derivative of Platform horizontal surge translation DOF (internal DOF ind...
48	ED First time derivative of Platform horizontal sway translation DOF (internal DOF ind...
49	ED First time derivative of Platform vertical heave translation DOF (internal DOF index...
50	ED First time derivative of Platform roll tilt rotation DOF (internal DOF index = DOF_R)...
51	ED First time derivative of Platform pitch tilt rotation DOF (internal DOF index = DOF_...
52	ED First time derivative of Platform yaw rotation DOF (internal DOF index = DOF_Y), r...
53	ED First time derivative of 1st tower fore-aft bending mode DOF (internal DOF index ...
54	ED First time derivative of 1st tower side-to-side bending mode DOF (internal DOF in...
55	ED First time derivative of 2nd tower fore-aft bending mode DOF (internal DOF index...
56	ED First time derivative of 2nd tower side-to-side bending mode DOF (internal DOF i...
57	ED First time derivative of Nacelle yaw DOF (internal DOF index = DOF_Yaw), rad/s
58	ED First time derivative of Variable speed generator DOF (internal DOF index = DOF_...
59	ED First time derivative of Drivetrain rotational-flexibility DOF (internal DOF index = D...
60	ED First time derivative of 1st flapwise bending-mode DOF of blade 1 (internal DOF i...
61	ED First time derivative of 1st flapwise bending-mode DOF of blade 2 (internal DOF i...
62	ED First time derivative of 1st flapwise bending-mode DOF of blade 3 (internal DOF i...
63	ED First time derivative of 1st edgewise bending-mode DOF of blade 1 (internal DOF ...
64	ED First time derivative of 1st edgewise bending-mode DOF of blade 2 (internal DOF ...
65	ED First time derivative of 1st edgewise bending-mode DOF of blade 3 (internal DOF ...
66	ED First time derivative of 2nd flapwise bending-mode DOF of blade 1 (internal DOF i...
67	ED First time derivative of 2nd flapwise bending-mode DOF of blade 2 (internal DOF i...
68	ED First time derivative of 2nd flapwise bending-mode DOF of blade 3 (internal DOF i...
69	SD First time derivative of Craig-Bampton mode 1 amplitude, -/s

E. Selecting Inputs and Outputs

The outputs and inputs of interest are in the same order as they are in the .lin files, so use those corresponding indices to choose your input (should be collective blade pitch, generator torque, wind and wave, and maybe cable tension in the future) and output variables.

Order of inputs from Model.lin file:

Order of inputs:

Column	Operating Point	Rotating Frame?	Derivative Order	Description
1	2.000000000E+001	F	0	IfW Extended input: horizontal wind speed (steady/
2	1.099999994E-001	F	0	IfW Extended input: vertical power-law shear expon
3	0.000000000E+000	F	0	IfW Extended input: propagation direction, rad
4	3.161157966E-001	T	0	ED Blade 1 pitch command, rad
5	3.161157966E-001	T	0	ED Blade 2 pitch command, rad
6	3.161157966E-001	T	0	ED Blade 3 pitch command, rad
7	0.000000000E+000	F	0	ED Yaw moment, Nm
8	1.998561094E+005	F	0	ED Generator torque, Nm
9	3.161157966E-001	F	0	ED Extended input: collective blade-pitch command,

Order of outputs from Model.lin file:

Order of outputs:

Row	Operating Point	Rotating Frame?	Derivative Order	Description
1	2.000000000E+001	F	0	IfW Wind1VelX, (m/s)
2	0.000000000E+000	F	0	IfW Wind1VelY, (m/s)
3	0.000000000E+000	F	0	IfW Wind1VelZ, (m/s)
4	1.998561249E+002	F	0	SrvD GenTq, (kN-m)
5	9.626104492E+003	F	0	SrvD GenPwr, (kW)
6	1.811210060E+001	T	0	ED BldPitch1, (deg)
7	1.811210060E+001	T	0	ED BldPitch2, (deg)
8	1.811210060E+001	T	0	ED BldPitch3, (deg)
9	3.095148010E+002	F	0	ED Azimuth, (deg)
10	9.582156181E+000	F	0	ED RotSpeed, (rpm)
11	4.791078186E+002	F	0	ED GenSpeed, (rpm)
12	0.000000000E+000	F	0	ED NacYaw, (deg)

F. Putting it all together to get your final state space matrices

Pick the indices of the inputs, states, and outputs to extract the rows and columns of interest from AvgA, AvgB, AvgC, AvgD.

```
A=AAvg(state_idx,state_idx);
B=BAvg(state_idx,input_idx);
C=CAvg(output_idx,state_idx);
D=DAvg(output_idx,input_idx);
```

IIII. Verification of the Linearization

Compare to nonlinear model using Simulink. However, this work is pending.

Step 1. Update ElastoDyn initial conditions for rotor speed and platform motion from average of post-mbc outputs.
Step 2. Make sure inflow wind has the correct steady wind speed.
Step 3. Get the avg GenTq and BldPitch from post-run mbc input values. Use these as inputs to the nonlinear model. Make sure in ServoDyn that PCMode and VSCntrl both equal 4.
Step 4. Run nonlinear model.
Step 5. Run motion adjusted wind speed calculation code, save this as a .mat file. ?
Step 6. Use the same GenTq and BldPitch input into the linear model. Put the motion adjusted wind speed as an input to the linear model.
Step 7. Run linear model.

Step 8. Compare outputs from nonlinear and linear model. Make sure GenTorque and BladePitch are basically equal and initial conditions are the same!