Special Assignment 1
ECES 511

Damien Prieur

This special assignment is designed for linearization and simulation of a single link planar arm with a DC motor. The model equations which were given in the class:

$$\begin{aligned}\dot{x}_1 &= x_2\\ \dot{x}_2 &= \frac{1}{J}(Nk_t x_3 - bx_2 - mgl\sin(x_1))\\ \dot{x}_3 &= \frac{1}{L}(u(t) - Rx_3 - k_b N x_2)\end{aligned}$$

where

$$\begin{bmatrix}x_1\\x_2\\x_3\end{bmatrix} = \begin{bmatrix}\theta\\\dot{\theta}\\i\end{bmatrix}$$

Input $u(t)$ to the system is the motor supply voltage.

The System parameters,

$$J = 0.02, k_t = k_b = 0.01, N = 10, R = 2, L = 0.5, b = 0.2, m = 1, g = 10, l = 0.3$$

a) Derive equilibrium states given input voltage $u = 15$, $u = 21.2$, $u = 30$,$u = 40$, and $u = 75$. What do you observe? Does the equilibrium exist for each input? Justify your answers.

We can find the equilibrium state for any input $u(t) =$ constant by setting all the derivatives to 0.

$$\begin{aligned}0 = \dot{x}_1 &= x_2\\ 0 = \dot{x}_2 &= \frac{1}{J}(Nk_t x_3 - bx_2 - mgl\sin(x_1))\\ 0 = \dot{x}_3 &= \frac{1}{L}(u(t) - Rx_3 - k_b N x_2)\end{aligned}$$

By the first equation we can see that $x_2 = 0$. Substituting we get

$$\begin{aligned}0 = \dot{x}_2 &= \frac{1}{J}(Nk_t x_3 - mgl\sin(x_1))\\ 0 = \dot{x}_3 &= \frac{1}{L}(u(t) - Rx_3)\end{aligned}$$

By the third equation we have $x_3 = \frac{u(t)}{R}$ Substituting we get

$$0 = \dot{x}_2 = \frac{1}{J}(Nk_t\frac{u(t)}{R} - mgl\sin(x_1))$$

By the second equation we get

$$\sin(x_1) = \frac{Nk_t u(t)}{Rmgl}$$

Combining these equations we get that equilibria for $u(t) = c$ for $c \in \mathbb{R}$

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} \sin^{-1}(\frac{Nk_t u(t)}{Rmgl}) \\ 0 \\ \frac{u(t)}{R} \end{bmatrix}$$

Plugging in for each value of u we get

$$u(t) = 15 \implies x \approx \begin{bmatrix} 0.25268 \\ 0 \\ 7.5 \end{bmatrix} \qquad u(t) = 21.2 \implies x \approx \begin{bmatrix} 0.361132 \\ 0 \\ 10.6 \end{bmatrix}$$

$$u(t) = 30 \implies x \approx \begin{bmatrix} 0.523599 \\ 0 \\ 15 \end{bmatrix} \qquad u(t) = 40 \implies x \approx \begin{bmatrix} 0.729728 \\ 0 \\ 20 \end{bmatrix}$$

$$u(t) = 75 \implies x \approx \begin{bmatrix} 1.5708 - 0.693147i \\ 0 \\ 32.5 \end{bmatrix}$$

In general as $u(t)$ increases so does the equilibrium angle (shown in radians) and it never passes $\frac{\pi}{2}$. The current also linearly increases which we expect due to $V = IR$. An equilibrium point can be calculated for each, but if the equilibrium point doesn't fall in $[-\frac{\pi}{2}, \frac{\pi}{2}]$ then we get complex values. This is because no real feasible equilibrium exists, the arm will continue to spin. For the $u(t) = 75$ example the motor is providing more torque than the weight at the point of highest torque so it will never stop rotating.
Side-note: each solutions has a second equilibrium point (infinitely many since we have a circle) that is outside the $[-\frac{\pi}{2}, \frac{\pi}{2}]$ range. These points are semi-stable equilibrium as from one direction they attract (coming from points in the stable range) and repel points on the other side. This is consistent with what we expect if we look at the torque when the arm is above the horizontal axis.

b) Derive linear models around equilibrium points for $u = 15$ volts and $u = 45$ volts. This will give you two separate linearized systems which you will need to convert into state equations. Hint: Look up the ss command in MATLAB.

Performing linearization around an equilibrium point $x_0 = \begin{bmatrix} x_{01} \\ x_{02} \\ x_{03} \end{bmatrix}$ with input $u(t) = u_0$ we get

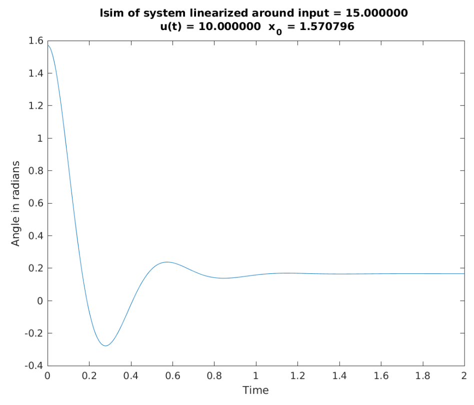
$$A = \frac{\partial h}{\partial x} = \begin{bmatrix} 0 & 1 & 0 \\ \frac{-mgl\cos x_1}{J} & \frac{-b}{J} & \frac{Nk_t}{J} \\ 0 & \frac{-k_b N}{L} & \frac{-R}{L} \end{bmatrix} \qquad B = \frac{\partial h}{\partial u} \begin{bmatrix} 0 \\ 0 \\ \frac{1}{L} \end{bmatrix} \qquad C = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \qquad D = \begin{bmatrix} 0 \end{bmatrix}$$

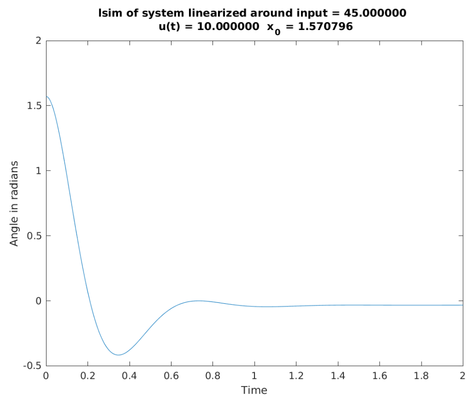
$$u(t) = 15 \qquad A = \begin{bmatrix} 0 & 1 & 0 \\ -145.2369 & -10 & 5 \\ 0 & -0.2 & -4 \end{bmatrix} \qquad \begin{bmatrix} 0 \\ 0 \\ 2 \end{bmatrix} \qquad C = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \qquad D = \begin{bmatrix} 0 \end{bmatrix}$$

$$u(t) = 45 \qquad A = \begin{bmatrix} 0 & 1 & 0 \\ -99.2157 & -10 & 5 \\ 0 & -0.2 & -4 \end{bmatrix} \qquad \begin{bmatrix} 0 \\ 0 \\ 2 \end{bmatrix} \qquad C = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \qquad D = \begin{bmatrix} 0 \end{bmatrix}$$

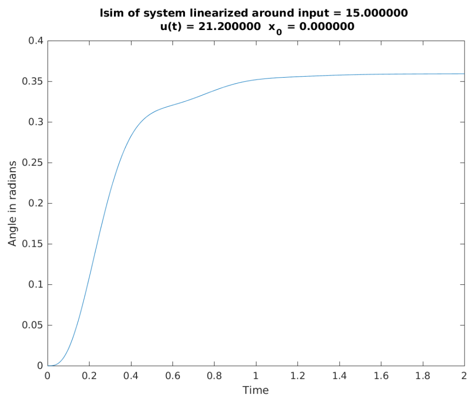
c) Simulate the linearized systems using MATLABS *lsim* command. Plot (you will need your state space models from part (b) for this) the state trajectories over a variety of different initial conditions for each of your linearized models.

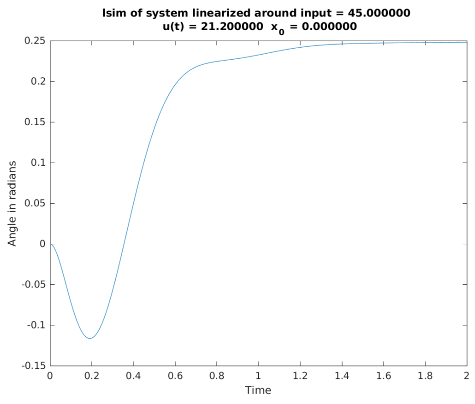
i. $u = 10, x_0 = \begin{bmatrix} \frac{\pi}{2} & 0 & 0 \end{bmatrix}$



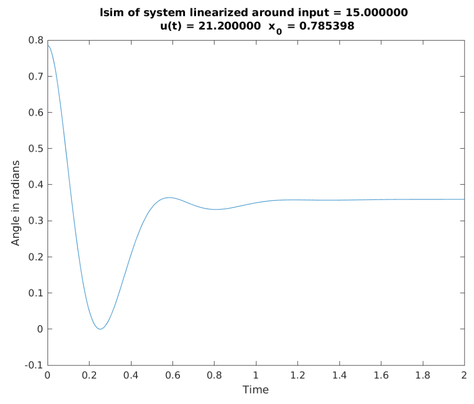


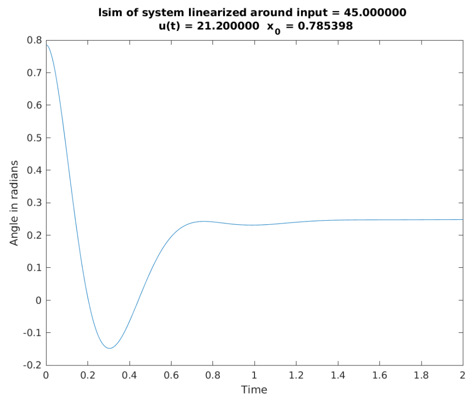
ii. $u = 21.2, x_0 = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}$



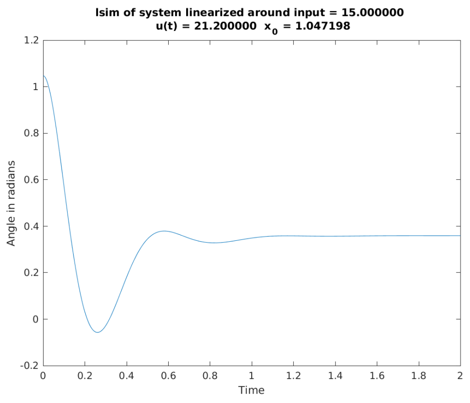


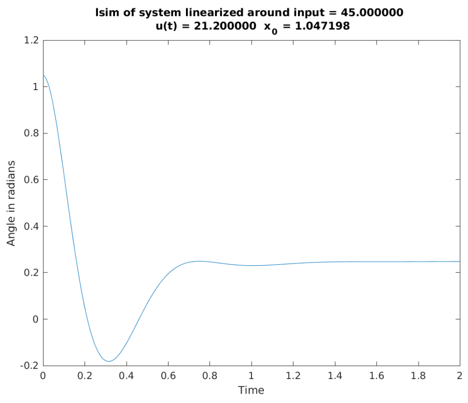
iii. $u = 21.2, x_0 = \begin{bmatrix} \frac{\pi}{4} & 0 & 0 \end{bmatrix}$



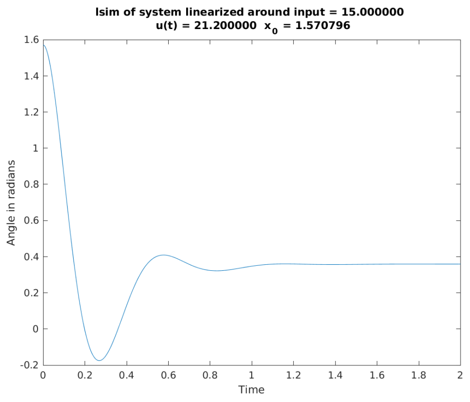


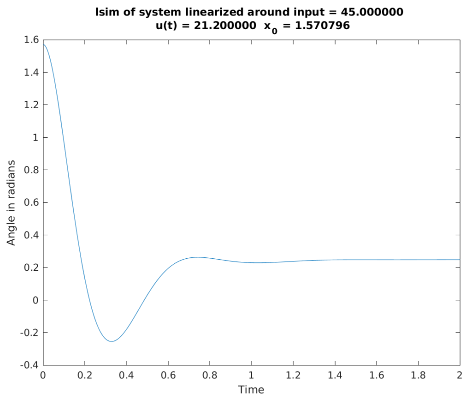
iv. $u = 21.2, x_0 = \begin{bmatrix} \frac{\pi}{3} & 0 & 0 \end{bmatrix}$



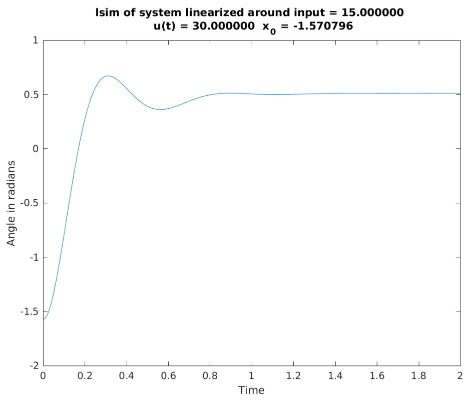


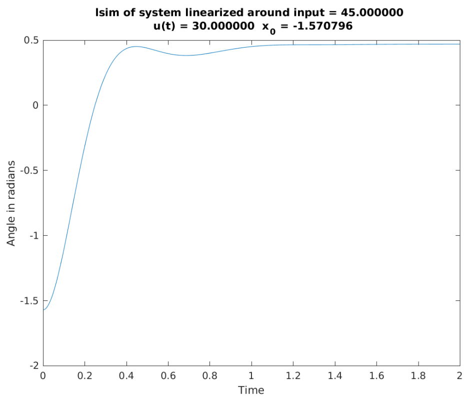
v. $u = 21.2, x_0 = \begin{bmatrix} \frac{\pi}{2} & 0 & 0 \end{bmatrix}$



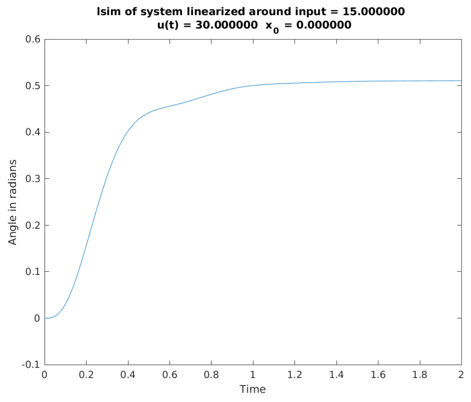


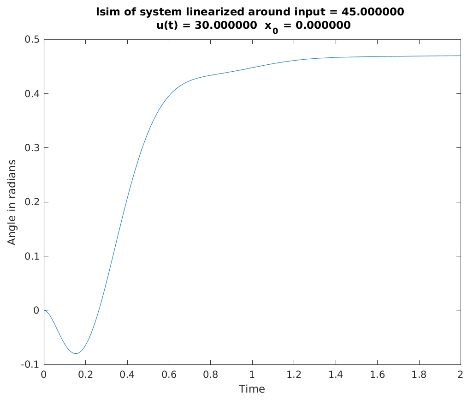
vi. $u = 30, x_0 = \begin{bmatrix} -\frac{\pi}{2} & 0 & 0 \end{bmatrix}$





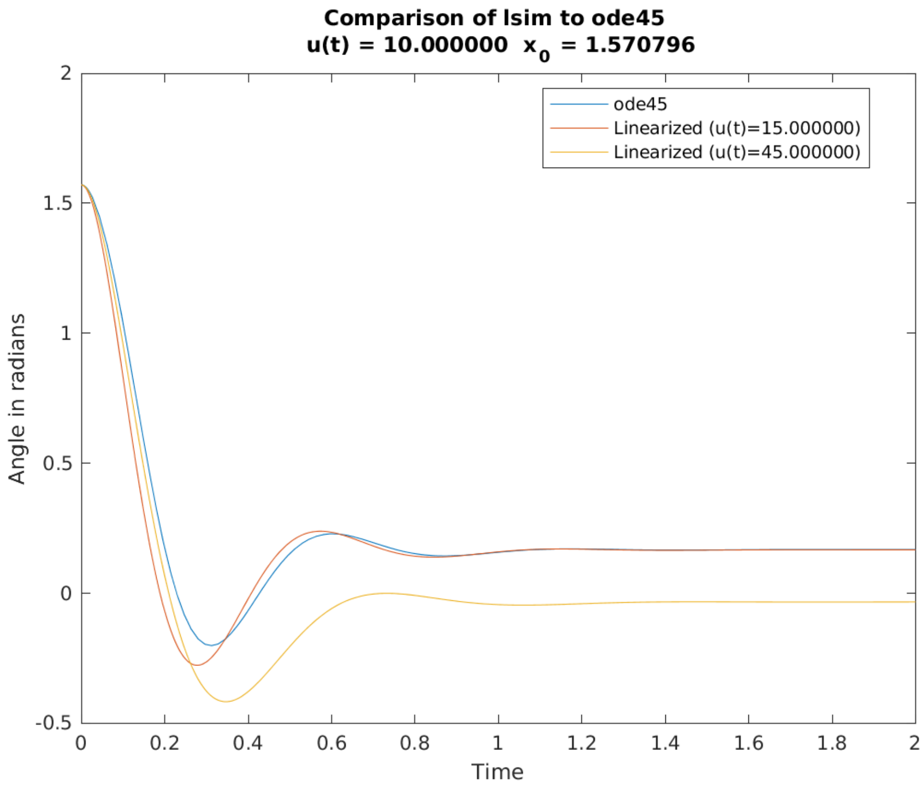
vii. $u = 30, x_0 = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}$



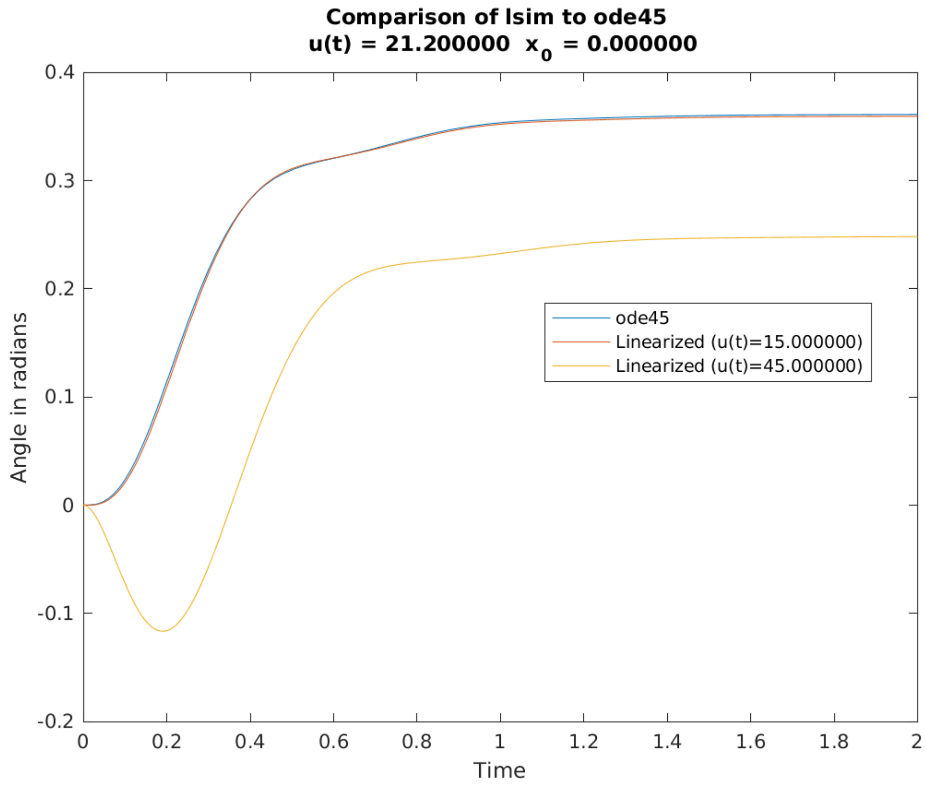


d) Repeat the exact same input and initial conditions used in part (c) but this time using the non-linear model and MATLAB's *ode*45 command. Compare your results from parts C and D by plotting the trajectories on the same figure. Play around with this and try different inputs with your three systems (non-linear and two linear models). How are each of the models behaving? For example, what happens if you use the 45 volt linearized model and give an input of 5 volts Vs 50 volts? What about 80 volts? Compare this to the non-linear model on the same plot?

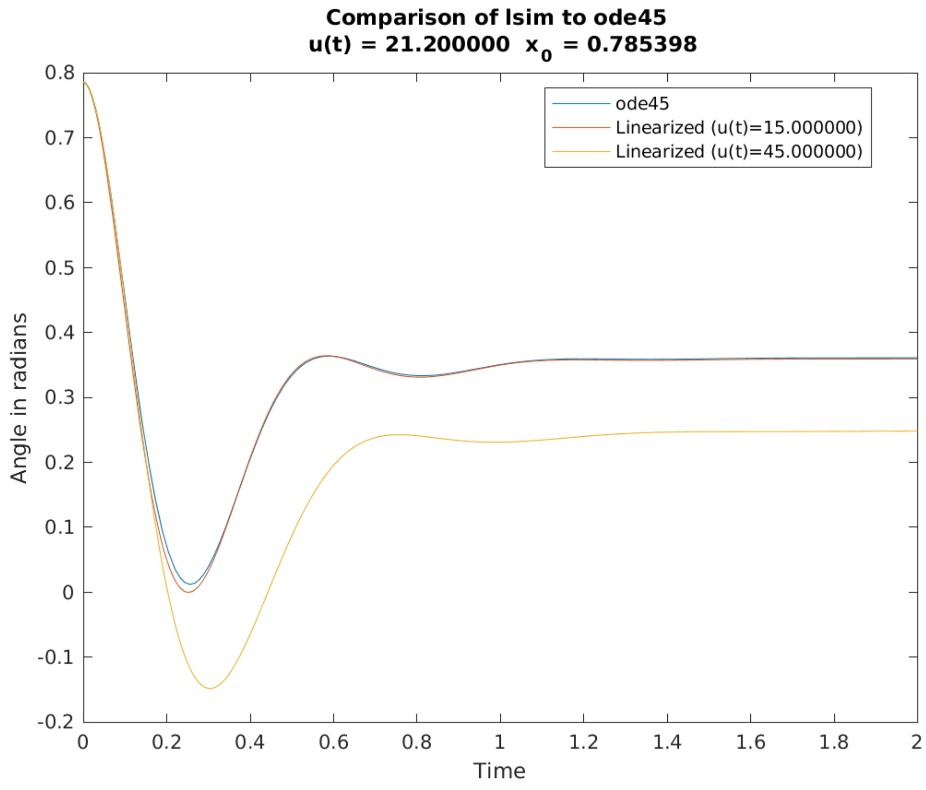
i. $u = 10, x_0 = \begin{bmatrix} \frac{\pi}{2} & 0 & 0 \end{bmatrix}$



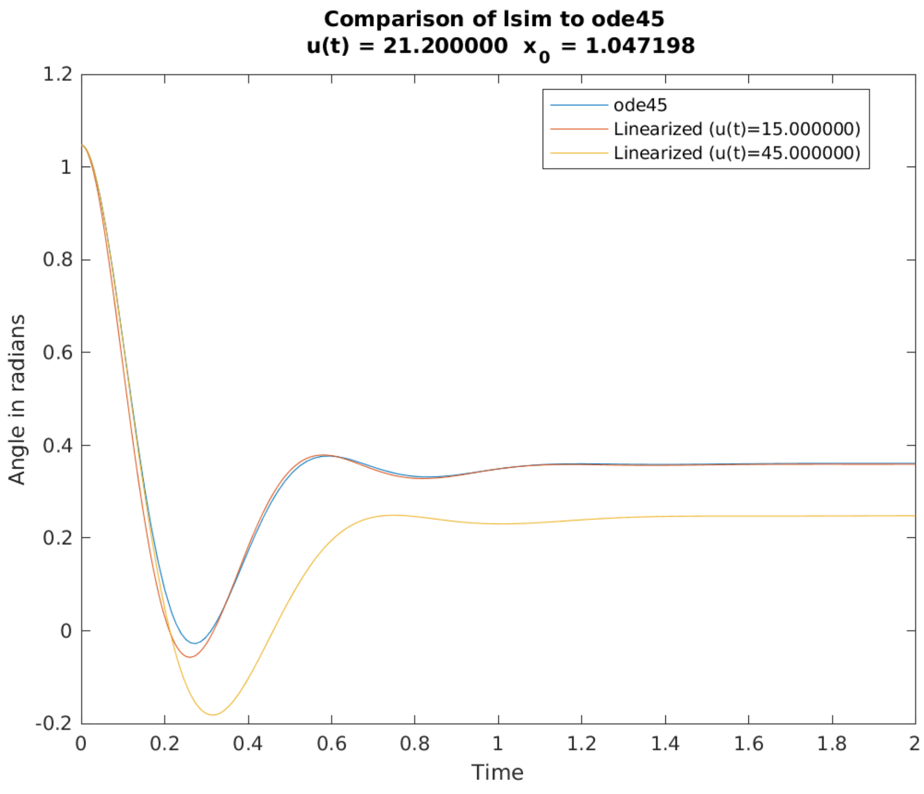
ii. $u = 21.2, x_0 = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}$



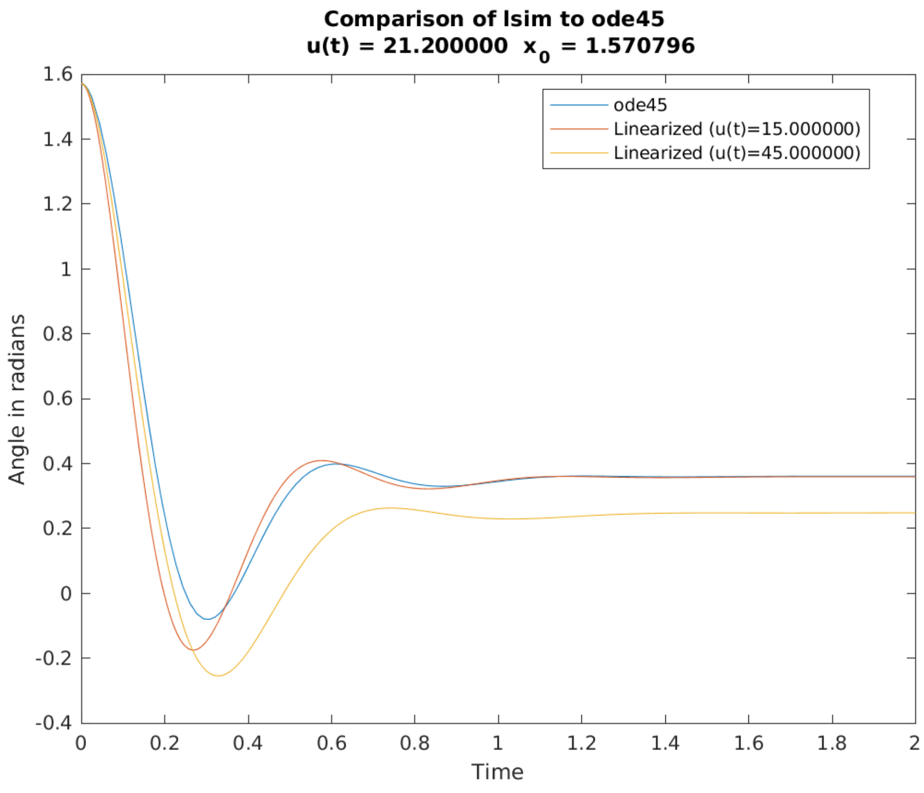
iii. $u = 21.2, x_0 = \begin{bmatrix} \frac{\pi}{4} & 0 & 0 \end{bmatrix}$



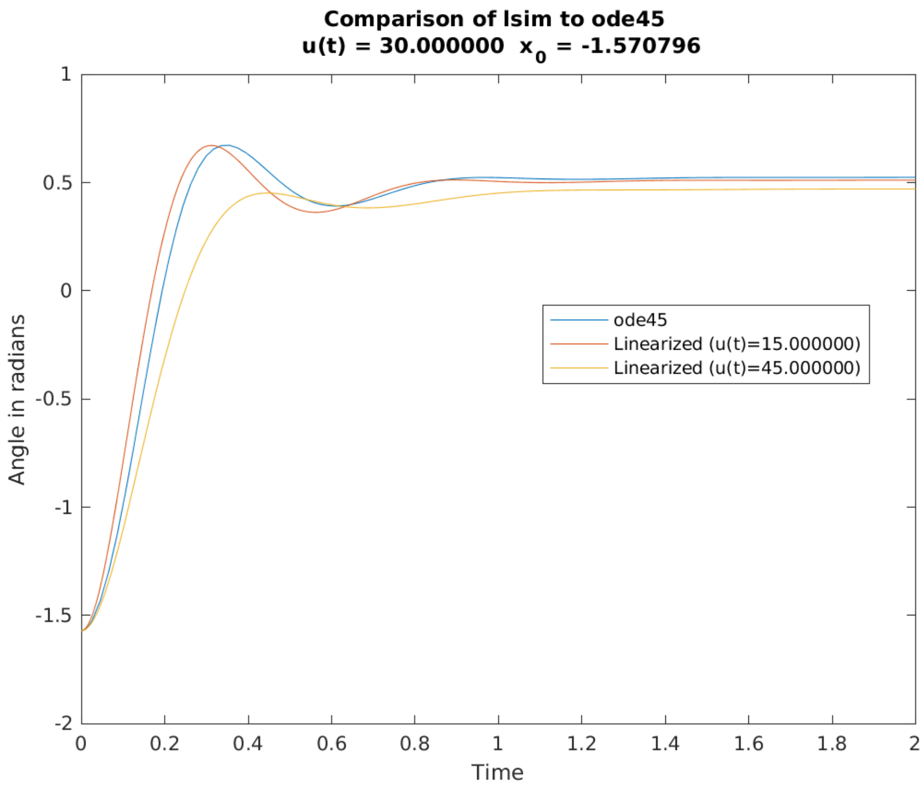
iv. $u = 21.2, x_0 = \begin{bmatrix} \frac{\pi}{3} & 0 & 0 \end{bmatrix}$



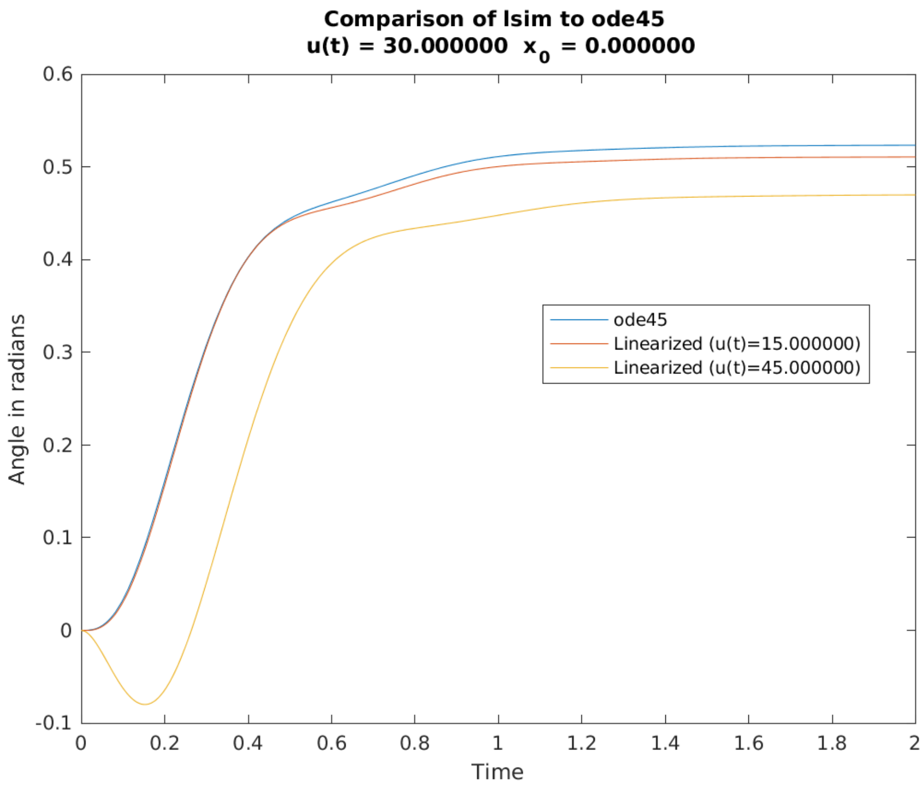
v. $u = 21.2, x_0 = \begin{bmatrix} \frac{\pi}{2} & 0 & 0 \end{bmatrix}$



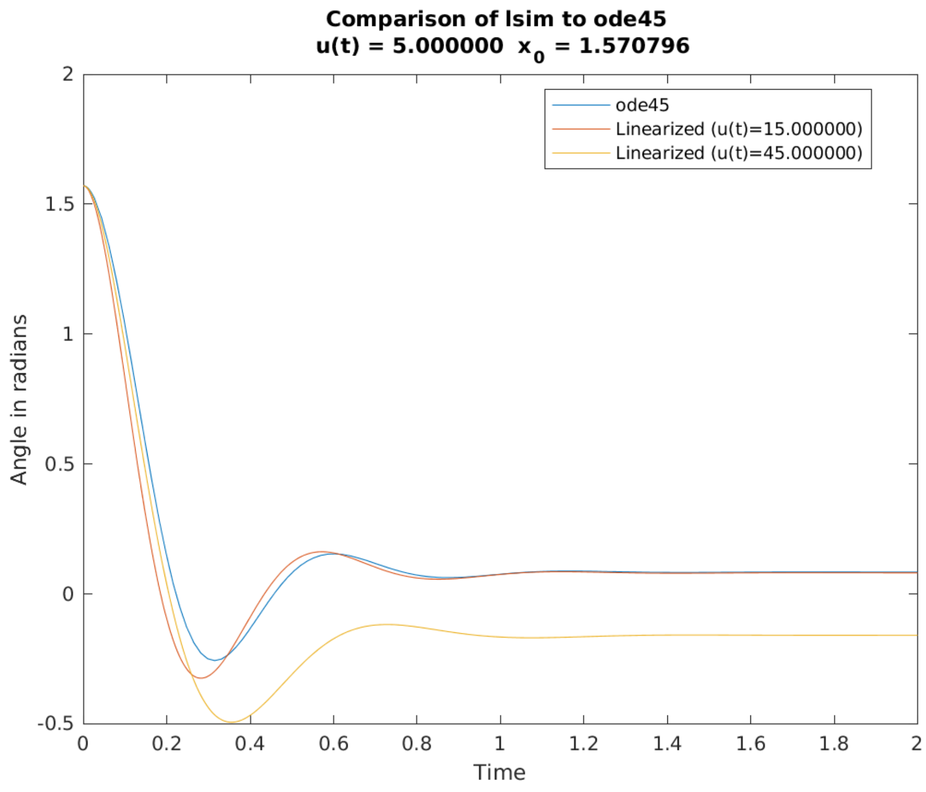
vi. $u = 30, x_0 = \begin{bmatrix} -\frac{\pi}{2} & 0 & 0 \end{bmatrix}$



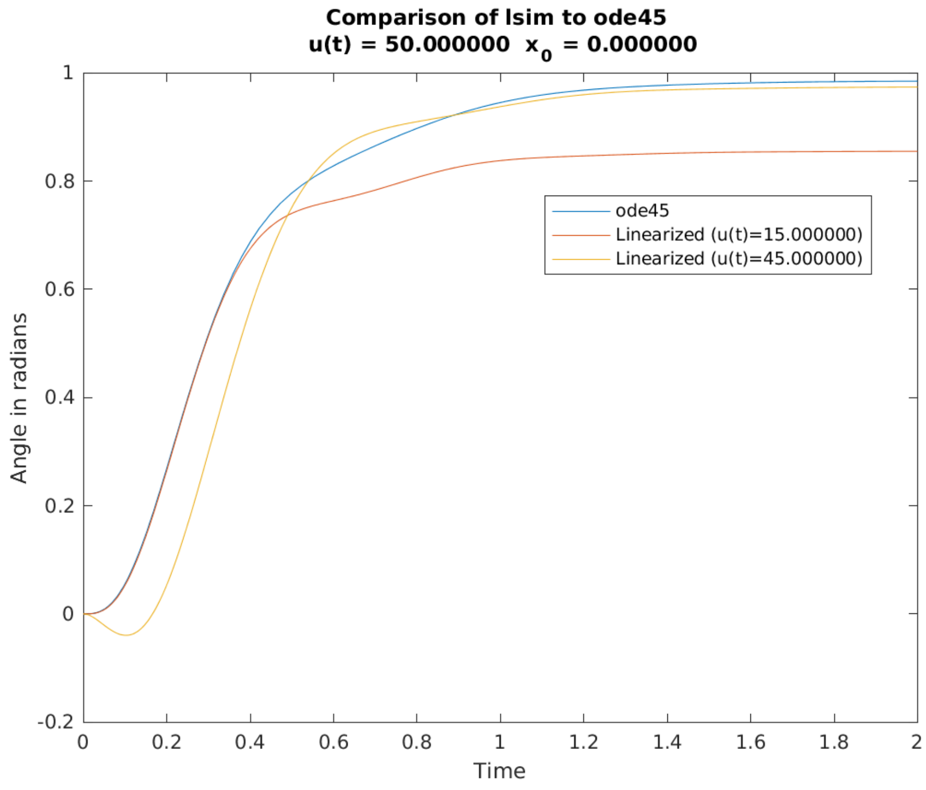
vii. $u = 30, x_0 = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}$



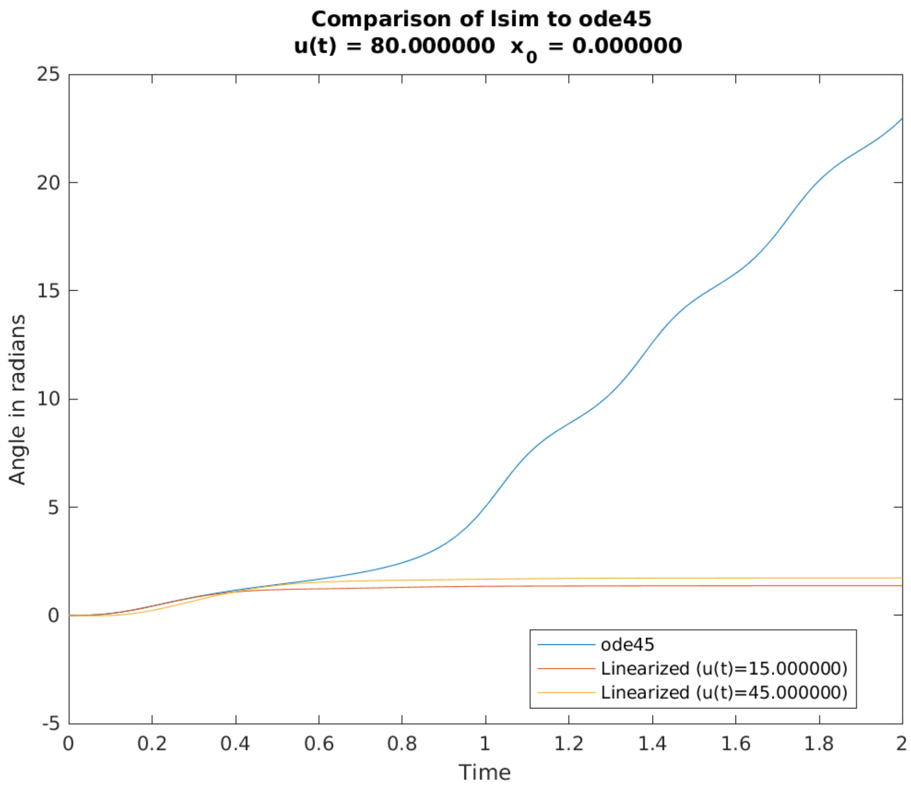
viii. $u = 5, x_0 = \begin{bmatrix} \frac{\pi}{2} & 0 & 0 \end{bmatrix}$



ix. $u = 50, x_0 = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}$



x. $u = 80, x_0 = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}$



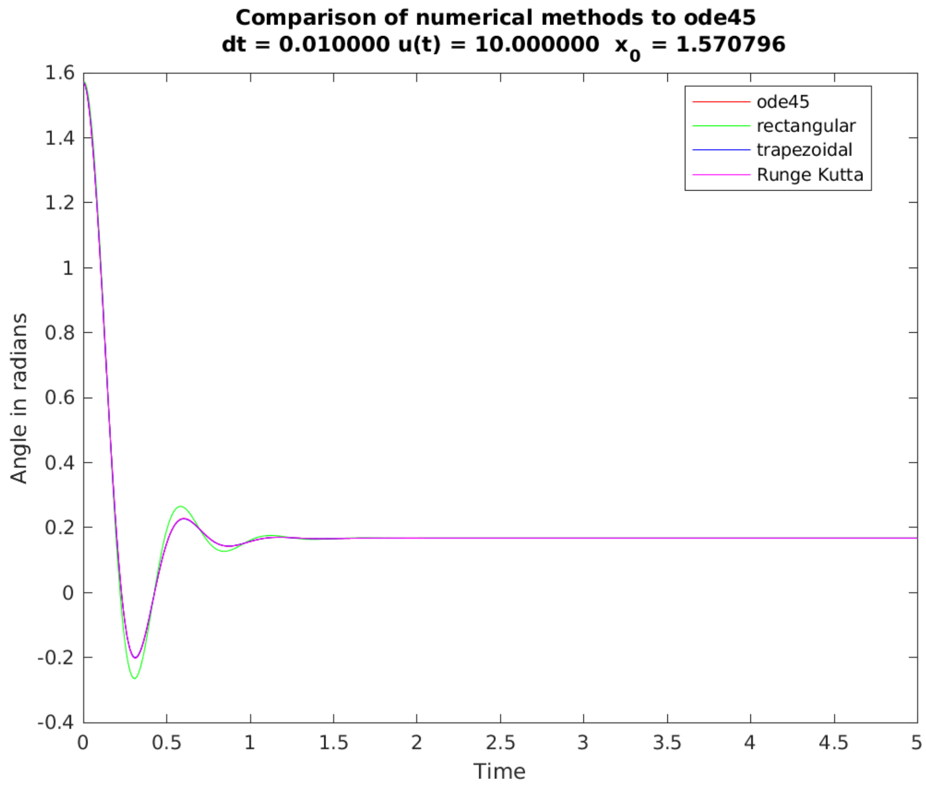
e) Compare (by plotting the trajectories on the same figure so that it is easy to compare) and discuss the results explaining the benefits and risks of the local linearization method.

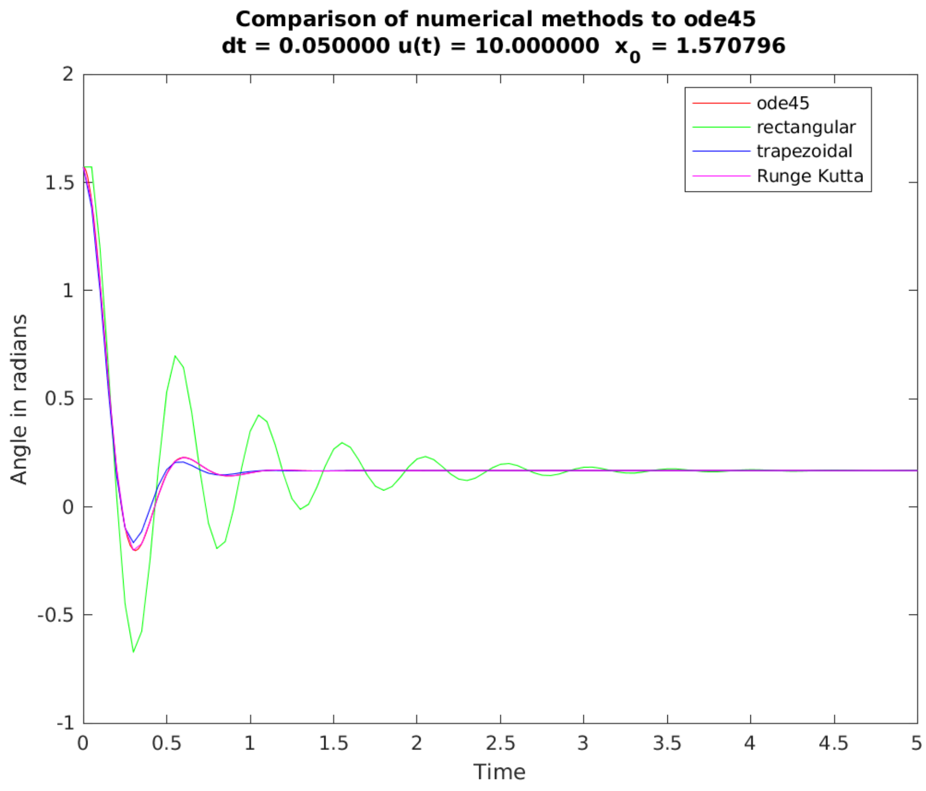
We can see that the linearization around $u(t) = 15$ is quite accurate for values of $u(t) < 30$. These values are accurate since the linearization around sin(x) is accurate for equilibrium points near $\theta = 0$. But as the equilibrium point for θ grows it becomes more difficult to use a linearization for a point far away from the actual equilibrium point. For initial positions far from the equilibrium point both models do poorly since the linearization can't capture the complexity.

Based on these results it seems to be safe to use a linearized model if the equilibrium point you linearize around is close to or the same as where you expect to operate the system.

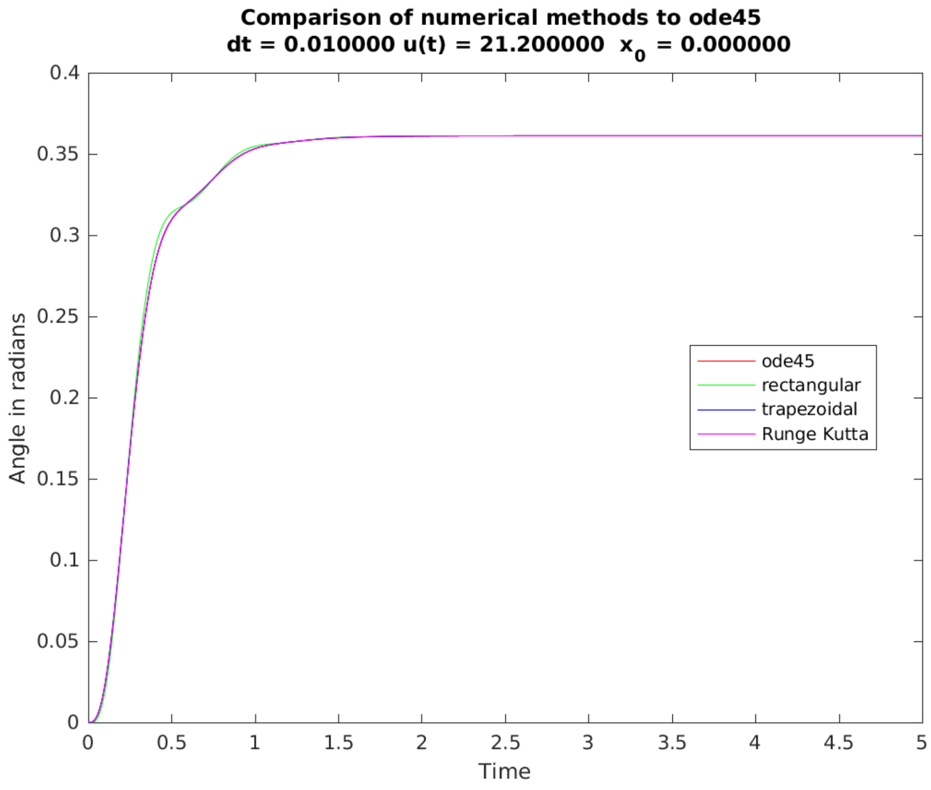
f) Repeat the exact same input and initial conditions simulation for 4-5 seconds duration using the Rectangular, Trapazoidal, and Runge Kutta numerical approximation techniques. Compare and plot all three techniques on the same graph Vs the nonlinear model using *ode*45.

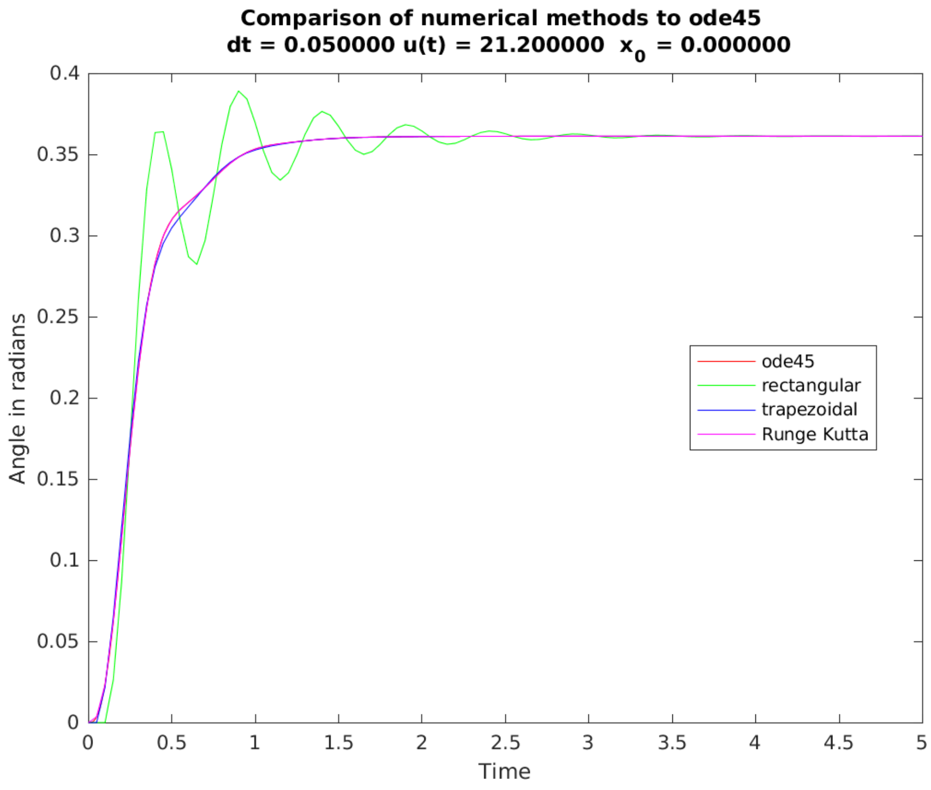
i. $u = 10, x_0 = \begin{bmatrix} \frac{\pi}{2} & 0 & 0 \end{bmatrix}$



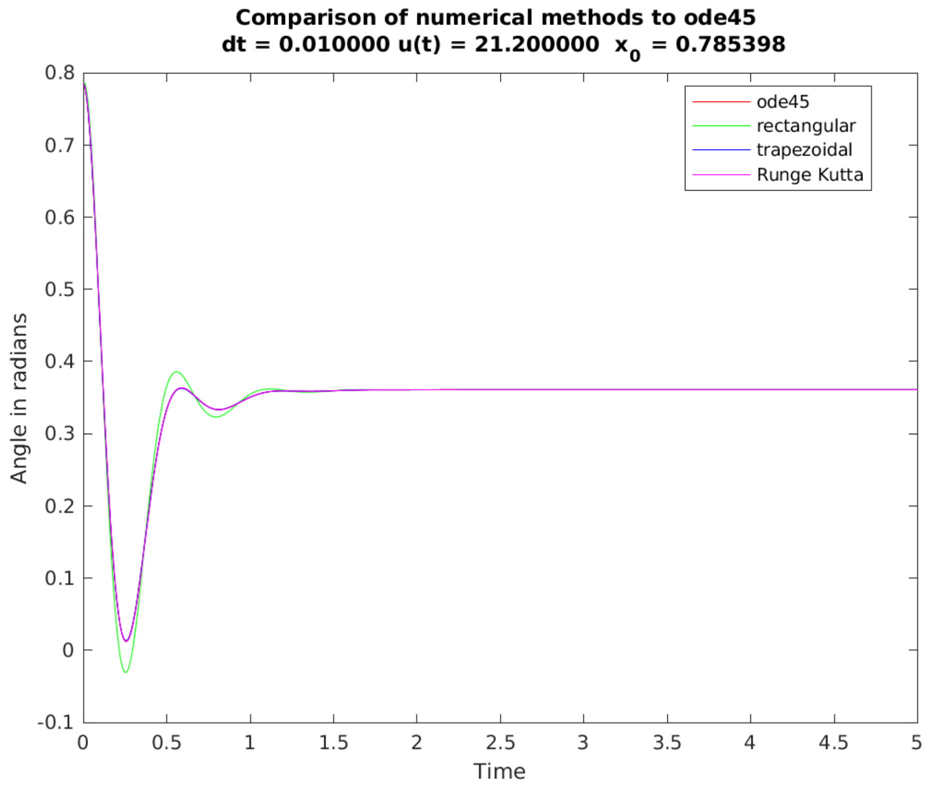


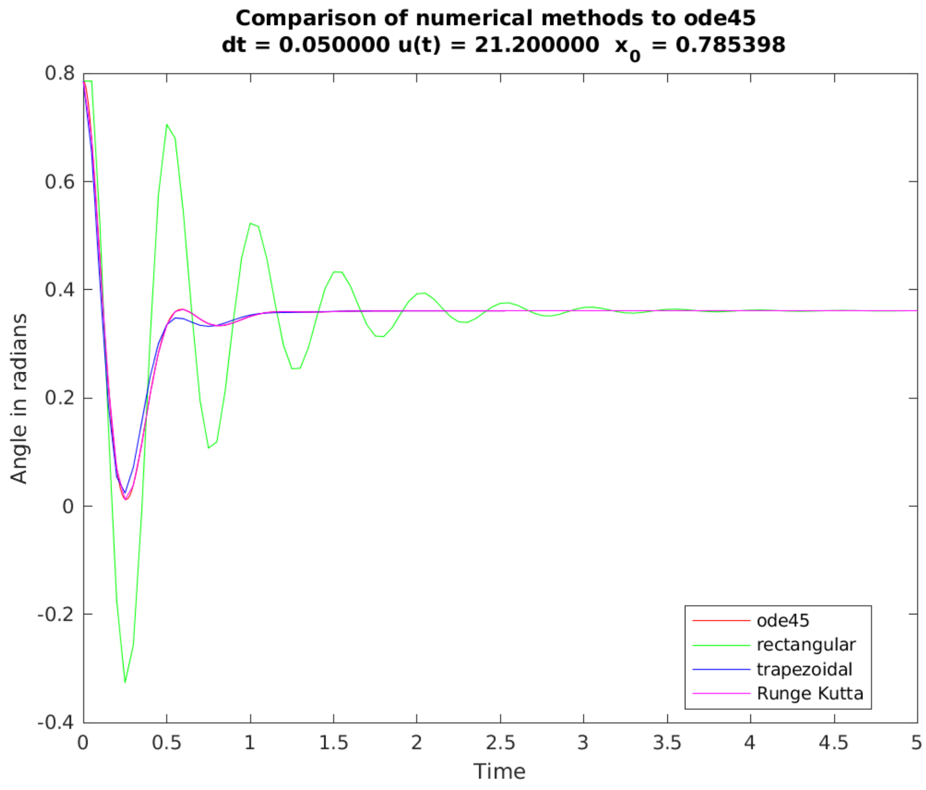
ii. $u = 21.2, x_0 = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}$



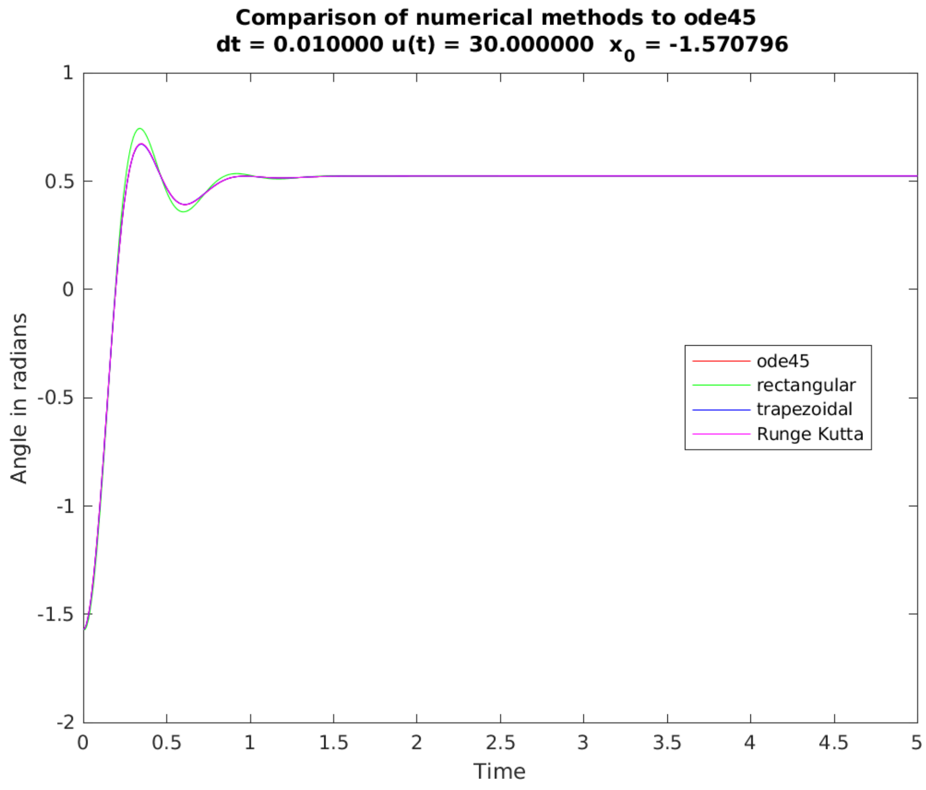


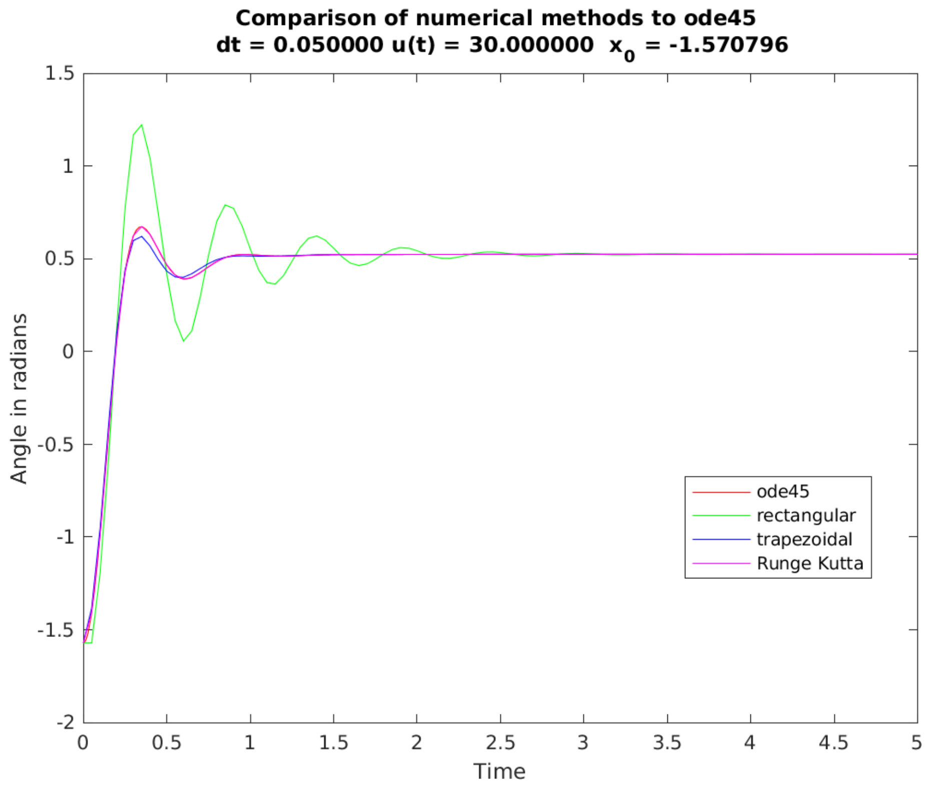
iii. $u = 21.2, x_0 = \begin{bmatrix} \frac{\pi}{4} & 0 & 0 \end{bmatrix}$



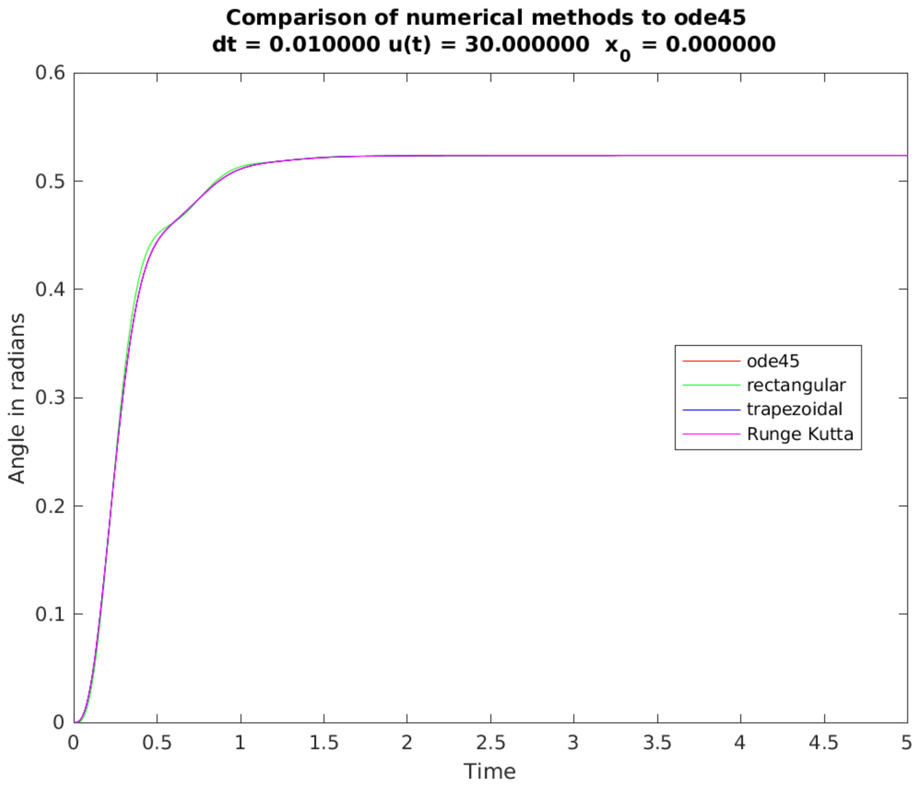


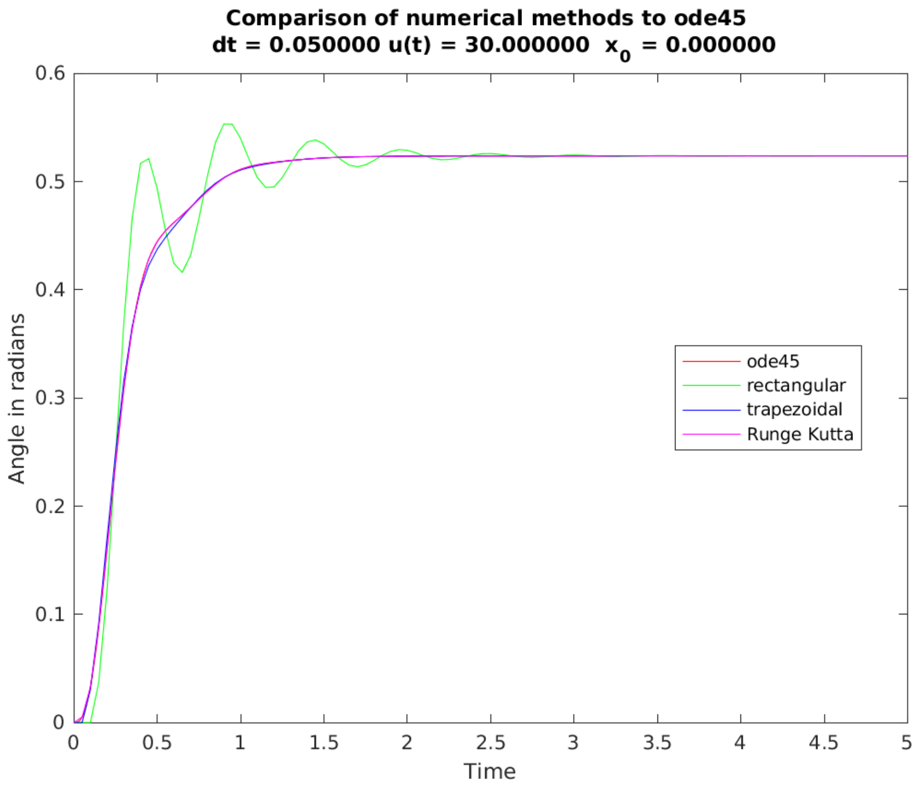
iv. $u = 30, x_0 = \begin{bmatrix} -\frac{\pi}{2} & 0 & 0 \end{bmatrix}$





v. $u = 30, x_0 = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}$





g) Compare (by plotting the trajectories on the same figure so that it is easy to compare) and discuss the results explaining the difference between numerical methods such as accuracy and efficiency.
We can see that using more datapoints, either decreasing Δt or using a method that uses more points, we get a better result. Rectangular integration is always the worst, but takes the fewest computations and is the easiest to implement. Trapezoidal gives a significantly better approximation even for relatively large Δt compared to rectangular with only twice the computational overhead. Meanwhile the Runge Kutta method give an extremely accurate approximation, but requires 4 times the computations as the rectangular method and twice as much as the trapezoidal. Given that computation is now not difficult to come by and is fast enough to perform these operations in realtime there is no reason why you shouldn't use the Runge Kutta method unless you are in a very computationally constrained system. One interesting thing is that all methods appear to converge to the same result regardless of the error from previous terms, but this is most likey due to the simplicity of our system and the relatively small Δt.