Logan Maier
Session 6

Wrapping GSL

1. The new version is better than the original because it seems to be more efficient. The new version is able to handle more types objects more appropriately, for example it now has support for Hamiltonian objects.
2. The code does still work
3. The Hamiltonian class does lack aspects that can analyze eigenvalues and eigenvectors.

OpenMP

1. My laptop is a dual core.
2. OpenMP is utilized in the K value calculations.
3. Run, cpu, and wall respectively are 3.245 sec, 6.684 sec, and 3.492 sec.
   Part 2 - Run, cpu, and wall respectively are 6.582 sec, 6.536 sec, and 7.281 sec.
   The parallelization is working as the run time is roughly doubled which makes sense since we are limiting the computation to run on one thread instead of spreading it across 2.
4. Run, cpu, and wall respectively are 1.968 sec, 7.954 sec, and 2.021 sec.
   1.968/3.245 = 0.6, which is supposed to be 0.5 when comparing 1 core to 2 core. The code does seem to scale decently but it isn't perfect. I think that it isnt perfectly scaling because we arnt performing this test in a vacuum. The processor has many other tasks that are using a portion of the cpu that is not dedicated to the program, whereas the scaling expects the entire cpu to be dedicated to it.

Integrating First-Ordered

3. RK4 appears to do much better with the exact values than the Euler method.
4. Y = -a*y*t is the differential being integrated.
5. Qualitatively the integration methods both reach round off error as expected. Rk4 reaches minimal error much sooner than Euler.
6. h^5 is the local error according to the graph
7. The graph for Euler error increases linearly, but the RK4 shows local minima and eventually reaches round-off error. For the RK4, I would use the graph to chose a step size. I would chose the smallest of the minima as the step size.