Investigating turbulence at UOIT's CLAIM lab was an experience that enriched my understanding of turbulence transition, programming, and numerical methods. The primary research project I pursued focused on 3D Kolmogorov transitional flow. Kolmogorov flow is defined as incompressible on a cubic domain with periodic boundary conditions and sinusoidal forcing in the streamwise direction [1]. Previous studies have computed several travelling wave solutions in 3D Kolmogorov flow and select edge (transitional) state dynamics have also been tracked and analyzed [1]. In Kolmogorov flow, there are three limit points corresponding to the laminar state, transition, and turbulence/chaotic state. The transitional state is called the edge state. While laminar and turbulent states are stable nodes in phase space, the edge state is a neutral saddle point, meaning it has a stable and unstable direction.

The goal for the first few weeks of the project was to track and find different edge states, given the known laminar and turbulent states, for a range of flow viscosity values between .07-.1 Pa s. It was important and necessary to compute edge states since they are the transitional states for Kolmogorov flow. Upon computing edge states, their dynamics could be studied to yield information on the mechanisms behind the flow's transition to turbulence or to relaminarize.

The code was written in Fortran 95, using MPI to parallelize the computations and decrease the run-time, and run on the multi-processer computers in UOIT's CLAIM lab. As the experience was done virtually, I obtained access to the lab's computers through an SSH protocol which allowed me to access the machines with a remote Linux server. The incompressible Navier-Stokes equations were solved using a pseudo-spectral code and using FFTs to compute the derivatives. The cubic domain size of 64x64x64 was used which was sufficient for refinement. The appropriate periodic boundary conditions and sinusoidal forcing terms were added to the equations to simulate Kolmogorov flow in the domain.

The code of most relevant interest integrated the Navier-Stokes equations and computed the edge states. A bisection method first described in Skufca et al. (2006) was written to carry out the edge tracking. The bisection method works by computing trajectories in time of the forcing component of the Fourier representation of the solution. This component was important to track because it corresponds to the analytical solution for 3D Kolmogorov flow, which is the forcing Fourier mode. First, the initial point of the forcing component was dependent on the bisection parameter, which determined where the trajectory would begin between the laminar and turbulent states. The initial conditions used for the forcing component were u=U+b*(u0-U), where b is the bisection parameter and u0 is an arbitrary flow state. The initial conditions were evolved in time until the trajectory approached the laminar or turbulent state.

Iteratively decreasing the bisection parameter bracketed the laminar-turbulent boundary, and thus it was possible to obtain a trajectory that evolved for a longer time without becoming laminar or turbulent. This corresponded to the forcing component following the edge for a finite time. Due to the limited numerical accuracy of the bisection parameter, it was not possible to remain on the edge for an infinite time. However, by restarting the bisection from the last closest trajectory point along the edge, it was possible to stay on the edge for an infinite time, and thus reach the edge state attractor. By computing edge states for various viscosity values, a limit point diagram could be created upon determining the solution type of the edge state. Limit point diagrams show the solutions of the system’s transition as a function of the parameter, viscosity.

Once an approximation to the edge state had been reached using the edge tracking method, it was crucial to identify the type of solution corresponding to the edge state. There are two qualitative ways to identify the edge state solution: through observation of the Fourier coefficients and reconstructing the vorticity field. The former involves plotting the time series of the other Fourier components and observing the trajectory behavior along the edge, before the trajectory became laminar or turbulent. The most common solutions observed were equilibria and travelling waves (which corresponded to a periodic orbit on the periodic domain). Equilibrium solutions were observed when the Fourier coefficients approached a constant value as t went to infinity after some initial transient motion. Sinusoidal wave patterns in the time series often signaled a travelling wave, particularly in Fourier coefficients whose direction aligned with the wave, usually in the streamwise (y) direction. If the orbit was found to have a shift, it indicated a modulated travelling wave.

To confirm the observed solution structure, the second approach of reconstructing the vorticity field was used. I wrote an integration code that selected the closest approximation to the edge state and integrated it more precisely with a smaller time step and for a total of one time unit. This data was read into a Matlab script which reconstructed the vorticity field of the flow for the edge approximation. Upon constructing the vorticity field, the data was read into Paraview to visualize the results in 3D. Isosurfaces were created for the streamwise direction to approximate the vortex surfaces, as well as counterrotating vortices. The data collected from the edge integration was shown through a series of frames displaying the vortex motion. The Paraview visualization gave an approximation of how the flow structures near the edge state behaved with time, which would confirm the type of solution. Most of the structures behaved as travelling waves near the edge state.

Sources

[1] Van Veen, L., and Goto, S., 2016, “Sub Critical Transition to Turbulence in Three-Dimensional Kolmogorov Flow,” Fluid Dyn. Res., **48** (2016), pp. 1-12.