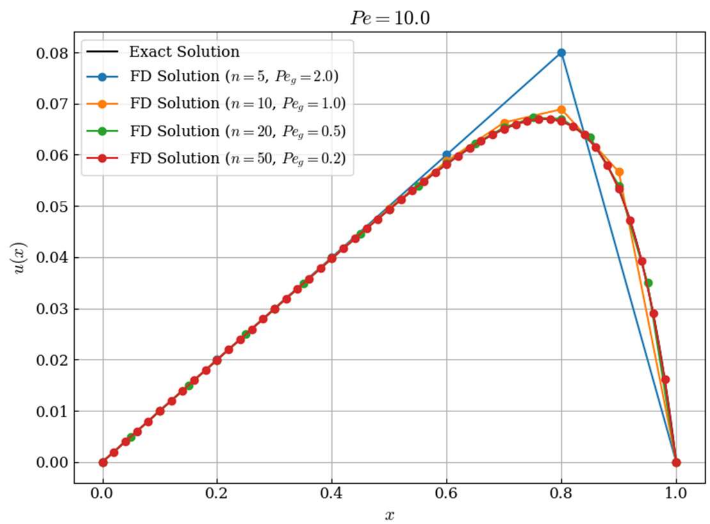
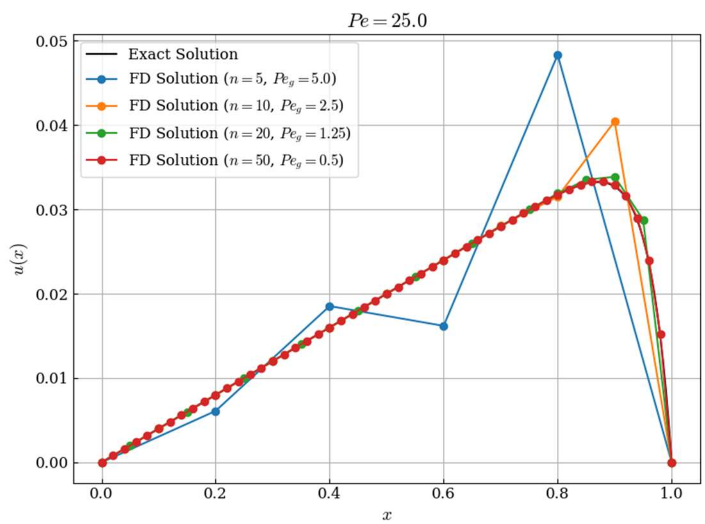
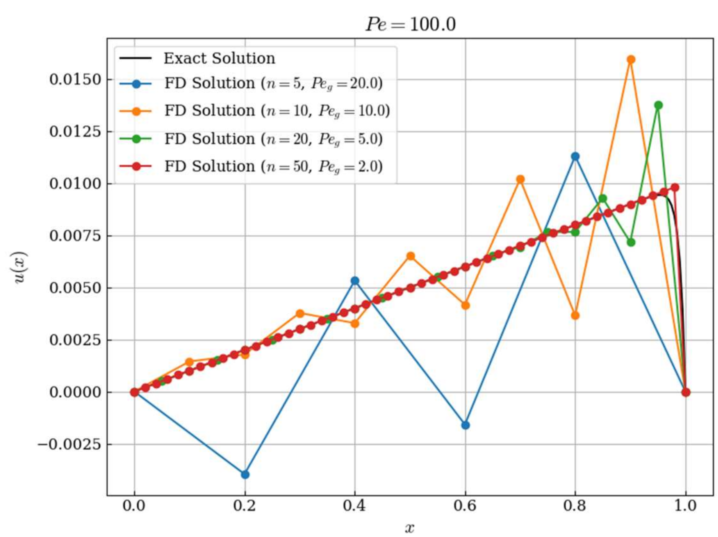
1.1


$Pe = 10.0$
Exact Solution
FD Solution ($n = 5$, $Pe_g = 2.0$)
FD Solution ($n = 10$, $Pe_g = 1.0$)
FD Solution ($n = 20$, $Pe_g = 0.5$)
FD Solution ($n = 50$, $Pe_g = 0.2$)
$u(x)$
x


$Pe = 25.0$
Exact Solution
FD Solution ($n = 5$, $Pe_g = 5.0$)
FD Solution ($n = 10$, $Pe_g = 2.5$)
FD Solution ($n = 20$, $Pe_g = 1.25$)
FD Solution ($n = 50$, $Pe_g = 0.5$)
$u(x)$
x

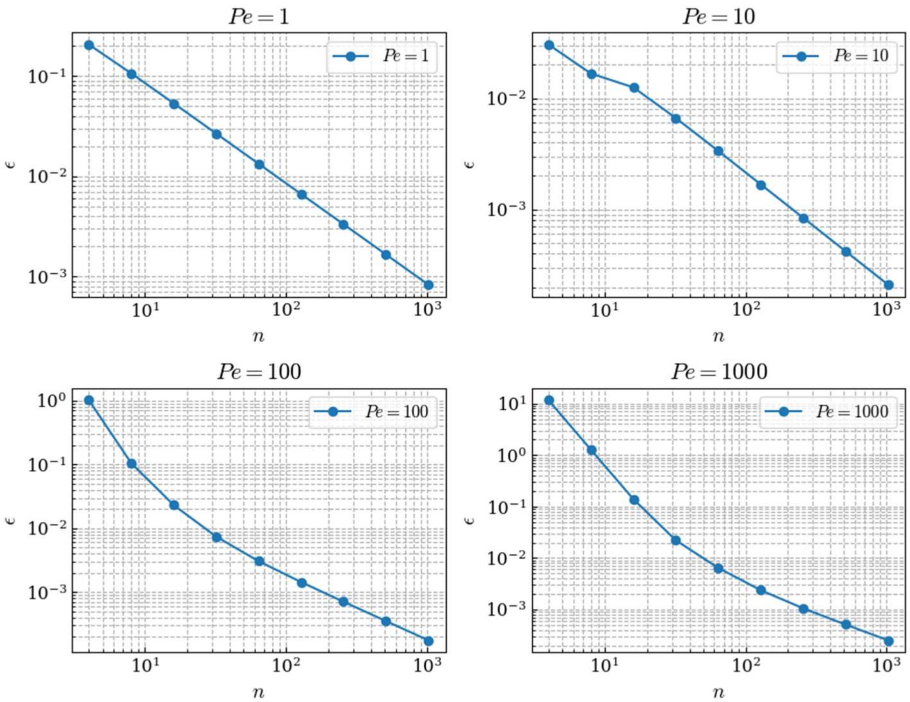

$Pe = 100.0$
Exact Solution
FD Solution ($n = 5$, $Pe_g = 20.0$)
FD Solution ($n = 10$, $Pe_g = 10.0$)
FD Solution ($n = 20$, $Pe_g = 5.0$)
FD Solution ($n = 50$, $Pe_g = 2.0$)
$u(x)$
x

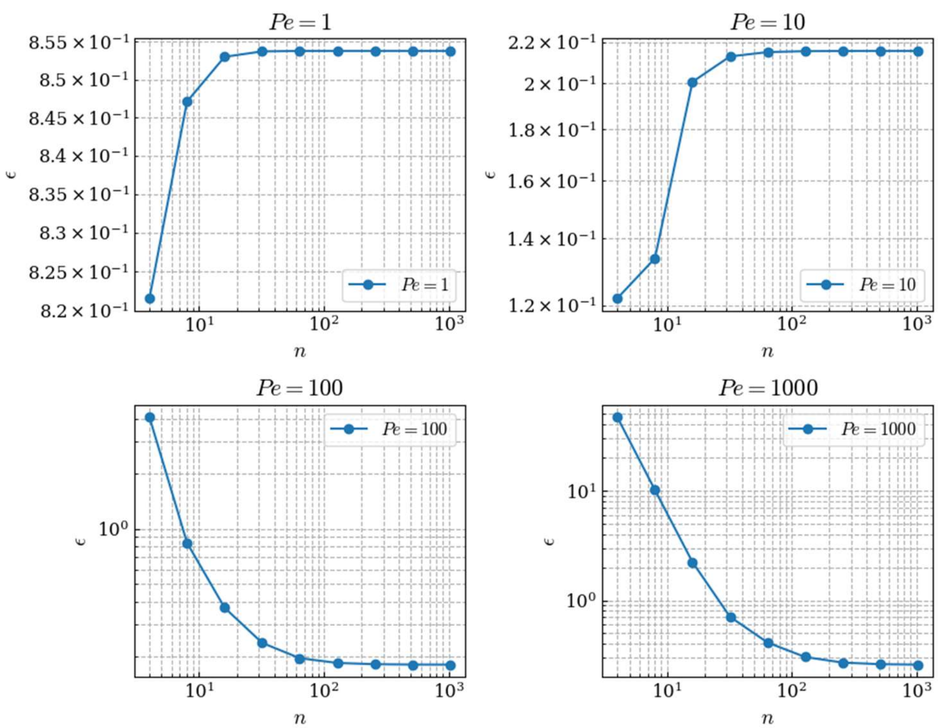
1.2

As discussed in the prompt, the oscillatory behavior is associated with the grid Peclet number, which is included in the legend of each plot. It is difficult to distinguish between the finite-difference method underresolving the exact solution and oscillations in some cases (n=50 for Pe=100), but the threshold for oscillations appears to be for $Pe_g = 50$.

1.3

The error can be normalized against the number of grid points or plotted as a total. Both plots are represented below.

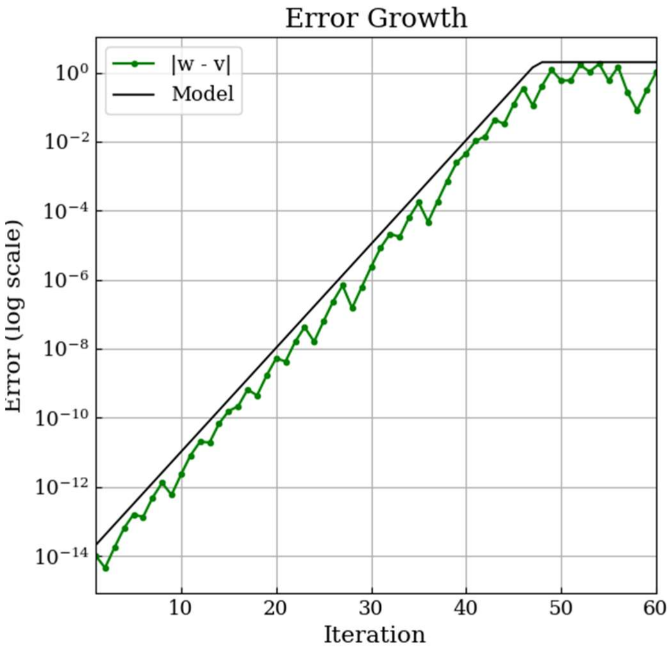


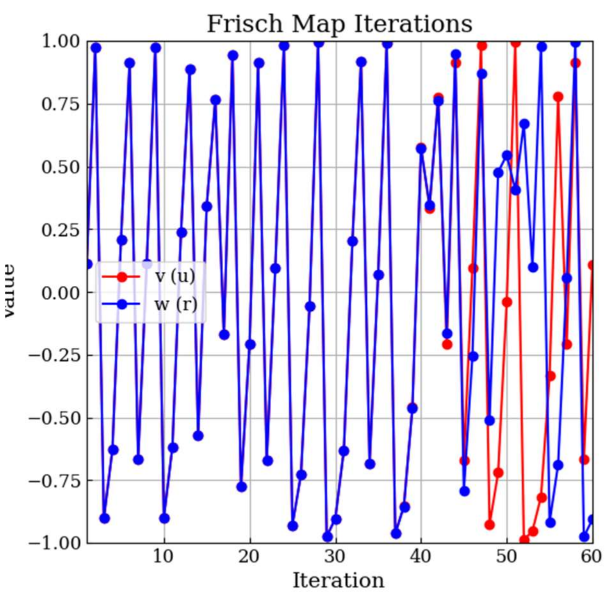


Normalizing the error (first plot) reveals a sub-linear rate of convergence with increasing Pe. This behavior is expected for a 1st-order scheme for which linear convergence is the best case. The total error (second plot) shows that convergence begins somewhere between Pe=10 and Pe=100, and the rate of convergence is nonlinear but sub-linear on average.

2.1

Redoing the analysis produces the following plots, after translating the matlab into python:





2.2 – what causes the model problem to be chaotic and how is NS similar?

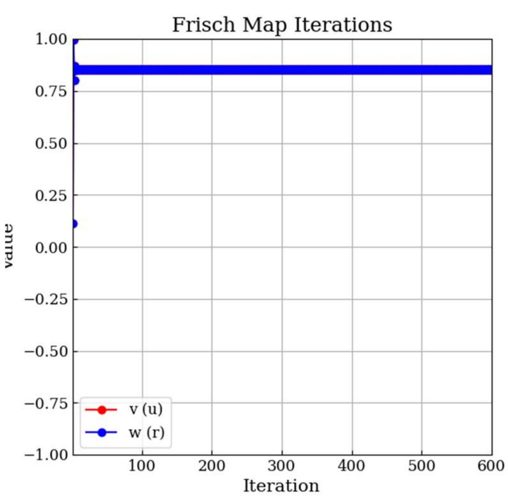
The v_n^2 term causes the model problem to be chaotic. It is easy to show that there is only one fixed point, $v^* = \frac{1}{2}$ which is unstable since $|f'(v^*)\} \geq 1$. Instability and boundedness are two requirements for chaos. Others are a lack of periodicity and a vast amount of possible trajectories. The Frisch map exhibits this.

The chaotic nature of the Navier-stokes equation comes from the nonlinear advection term. When advection is dominant (at high Re, diffusion is negligible) gradients are magnified on smaller scales in a fluid. Diffusion has a damping effect, and below a critical threshold the small differences are not damped and instead impact the flow. Mathematically, the nonlinear advection term behaves similarly to the v_n^2 term and magnifies small differences.

2.3 – how to modify to reduce the effects of chaos

An approach to modifying the map to reduce the chaotic effect is found in the similarity to NS. To reduce the chaos in a fluid flow, the effects of viscosity are increased by decreasing Re. Similarly, the multiplier on the chaotic term can be changed from -2 to -0.2, which completely eliminates the chaos. However, it also gets rid of the interesting behavior of the Frisch map.

The divergence plot is below, for 600 iterations to show that the order in the solution is not purely due to a lack of time for chaos to develop. The error plot is not included because the error is 0 across all n.



3.1

Roundoff error is caused by representing real numbers, which often require infinite digits to be expressed in decimal form, with finite digits. They are expressed as floating-point numbers with a significand and an exponent. Each floating-point number is an imperfect representation of a real number (excepting the rare cases where the base of the real number matches the floating point base).

In that sense, roundoff error exists in any floating point expression of a number. However, roundoff error typically refers to the buildup of those errors. When mathematical operations are performed on floats, more bits are needed to express the result of the operation than are available in a float. Therefore, a rounding is performed and a loss of precision occurs.

3.2

The conditioning of a mathematical operation refers to how sensitive the operation is to perturbations in initial data. As discussed in **3.1**, floating point operations necessarily introduce some perturbations through the loss of precision. If operations are being performed which are extremely sensitive to the perturbations (the most extreme of which would be looking only at the final bit in a float), roundoff errors magnify quickly.

3.3

Higher order methods are less subject to roundoff error because they require less steps, iterations, or grid points to come to a solution of the same accuracy as a low-order scheme. If we consider the numerical solution to an ODE (though without loss of generality this applies to PDEs or general equations), we can write:

$$\lim_{\Delta \mathrm{x} \to 0} |u(x) - u_N(x)| = 0$$

if the numerical scheme is correctly chosen. The error can be quantified as:

$$|u(x) - u_N(x)| \leq C \Delta x^n$$

Where C is a constant and n represents the order of accuracy of the scheme. To get to the same level of error, a low order scheme (small n) requires small Δx. A high order scheme (large n) can get to the same error with large Δx.

Therefore, for the same level of accuracy, a high order scheme needs less grid points. Roundoff error is introduced any time a floating point operation is performed. The high order scheme needs less grid points and therefore less floating point operations.

The same logic applies to temporal discretization or multi-dimensional spatial discretizations. Higher order schemes can use less degrees of freedom to get the same result, which causes them to have lower roundoff error.