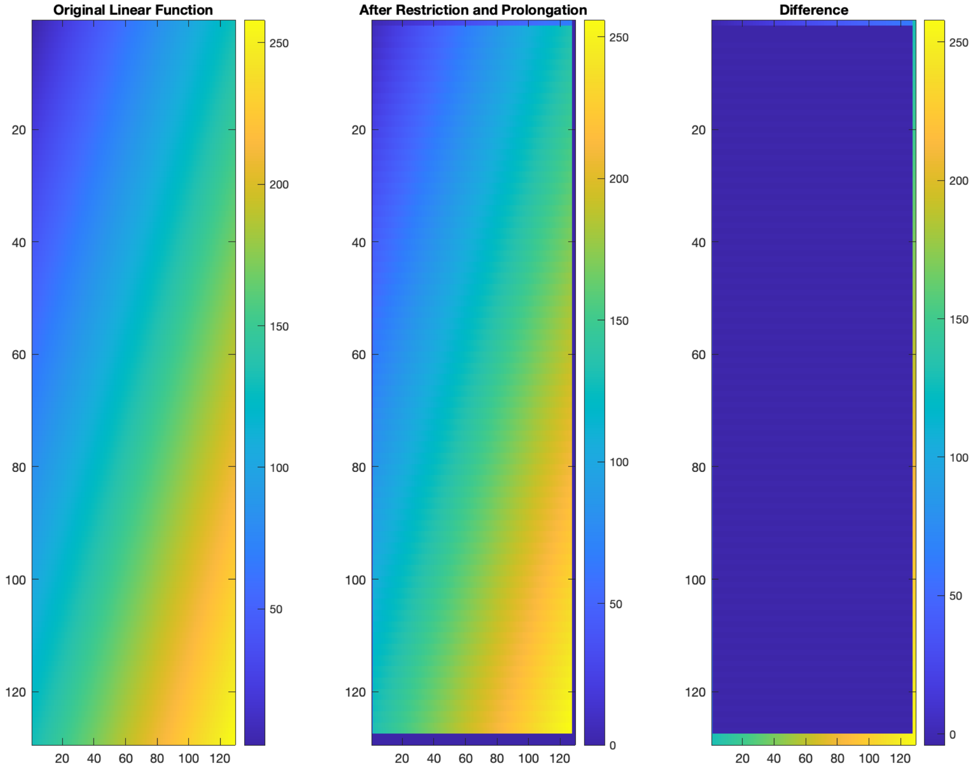
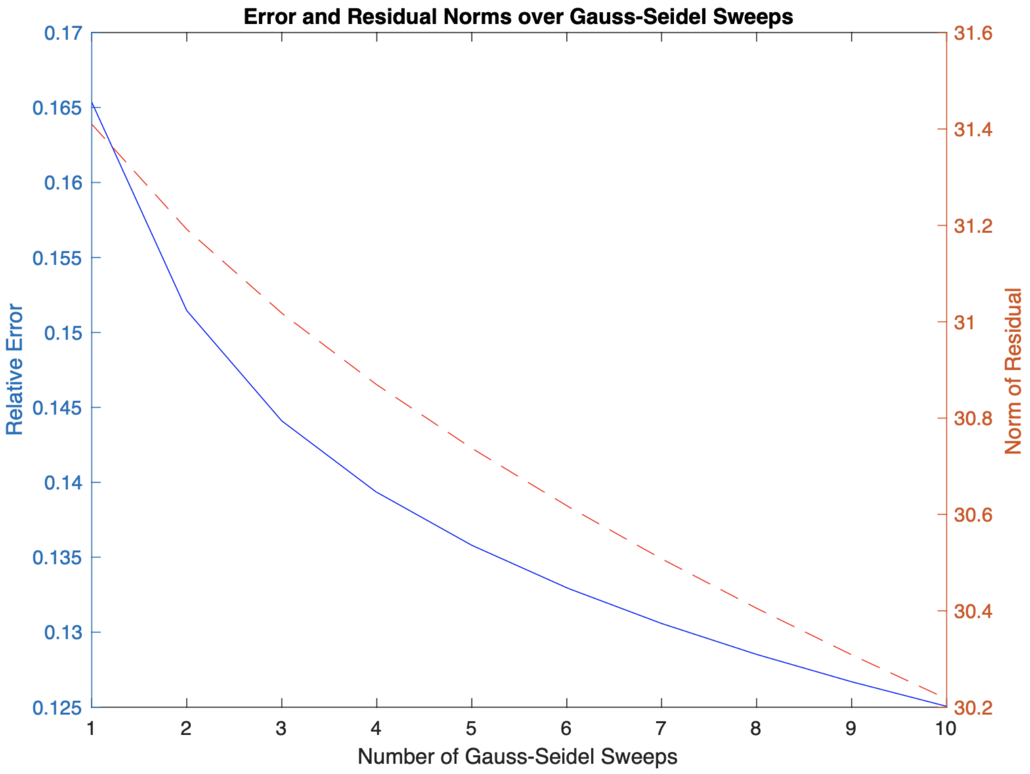
Multigrid - HW4:
Code attached here: https://github.com/lwyhasacat/HW4_Multipgrid

Exercise 1:

To check that the restriction and prolongation operators are coded correctly, we can use a test linear function, plot it before the restriction and prolongation, after the restriction and prolongation, and the difference between the two. We can see that the restrictions and prolongations of the test linear function produce the same linear function.

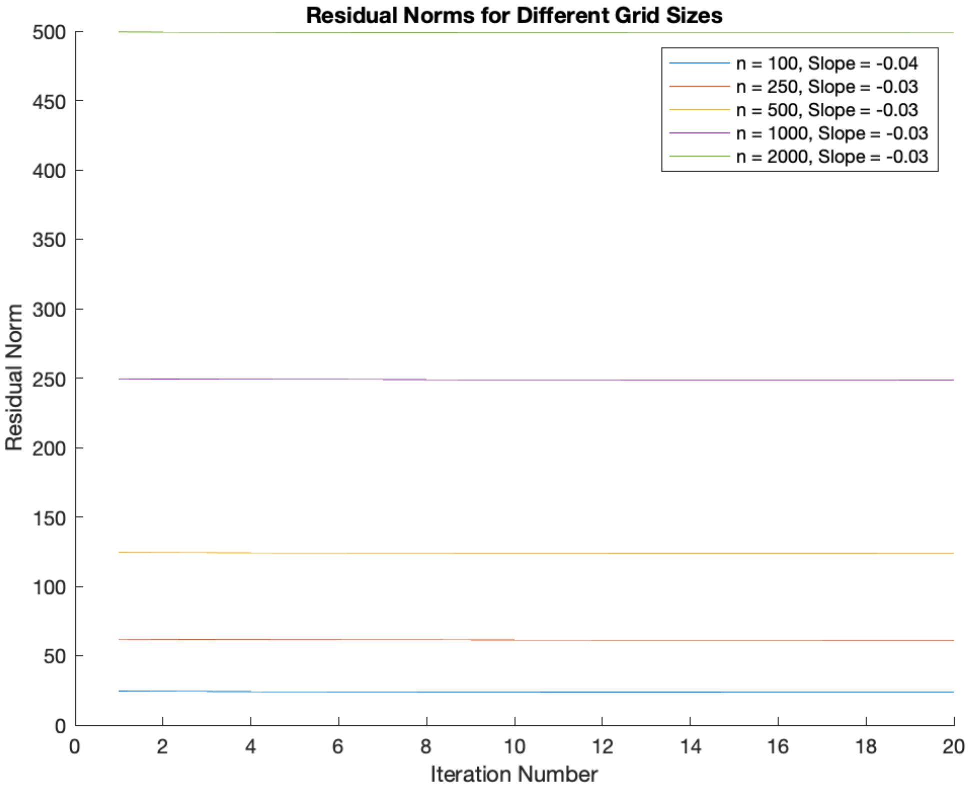


After using different scalings of the y-axis, we can see both curves clearly as expected. The error e(t) shown in blue decreases more rapidly than the residual r shown in red, which suggests that the error is smooth enough to be well-represented even on a coarse grid, meaning that if we use a coarse grid instead of a fine one where computations are cheaper, we will still be able to capture the smooth errors and correct it effectively.



Exercise 2:

Choosing p = 2 and varying n, we can see that the rate of decrease of the residual is independent of n, which means that the efficiency of the two-grid method in reducing errors (the rate of decrease of the residuals per iteration) is consistent regardless of the grid size.



Exercise 3:

On the finest grid with n^2 points, performing p pre-sweeps and r post-sweeps of Gauss-Seidel would each have a computational complexity of $O(n^2)$. The work at each subsequent coarser grid decreases geometrically because each coarser grid has about a quarter ($\frac{1}{4}$) of the points of its finer predecessor. Then, when we are calculating the total work, for a V-cycle, the work can be computed as:

$$W(n) = (p+r)(n^2 + \frac{n^2}{4} + \frac{n^2}{16} + \ldots)$$

This will converge and the sum is:

$$W(n) = (p+r) \cdot n^2 \cdot \left(\sum_{k=0}^{\infty} \frac{1}{4^k} \right) = (p+r) \cdot n^2 \cdot \frac{4}{3} = O(n^2)$$

For a W-cycle, we use more than one multigrid iterations on each coarser grid:

$$W(n) = (p+r) \cdot n^2 \cdot \left(1 + 2 \left(\frac{1}{4} + \frac{1}{16} + \ldots \right) \right)$$

We will get a finite sum for the inner series and therefore the total work is still $O(n^2)$.

For $q = 3$, three iterations are used on each coarser grid level, and the work at each level involves the recursive application of three multigrid operations:

$$W(n) = (p+r) \cdot n^2 \cdot \left(1 + 3 \left(\frac{1}{4} + 9\frac{1}{16} + 27\frac{1}{64} + \ldots \right) \right)$$

which is a geometric series with a ratio of $\frac{3}{4}$ and the sum would be:

$$\sum_{k=0}^{\infty} 3^k \left(\frac{1}{4} \right)^k = \frac{1}{1 - \frac{3}{4}} = 4$$

Thus, the total work for $q = 3$ is still $O(n^2)$ as:

$$W(n) = (p+r) \cdot n^2 \cdot 4 = O(n^2)$$

For $q = 4$, similarly:

$$W(n) = (p + r) \cdot n^2 \cdot \left(1 + 4\left(\frac{1}{4} + 16\frac{1}{16} + 64\frac{1}{64} + \dots\right)\right)$$

which takes the form:

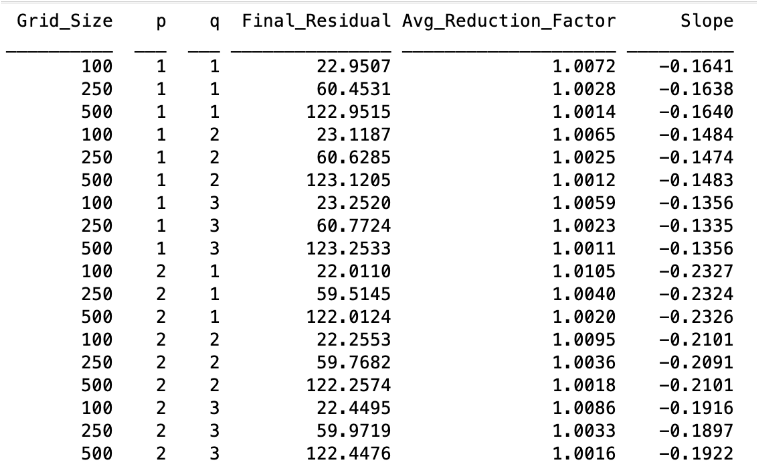
$$\sum_{k=0}^{\infty} 4^k \left(\frac{1}{4}\right)^k = \sum_{k=0}^{\infty} 1^k = \infty$$

It doesn't converge, but when we are actually using the multigrid method, the recursion is only applied up to a practical limit and the actual number of operations will add up to a limit that increases like a logarithmic progression with n similar to traversing a binary tree (in this case the depth of the tree would be the recursion depth, which increases logarithmically with the number of grid points). Then, we have:

$$W(n) = (p + r) \cdot n^2 \cdot \log(n) = O(n^2 \log(n))$$

Exercise 4:

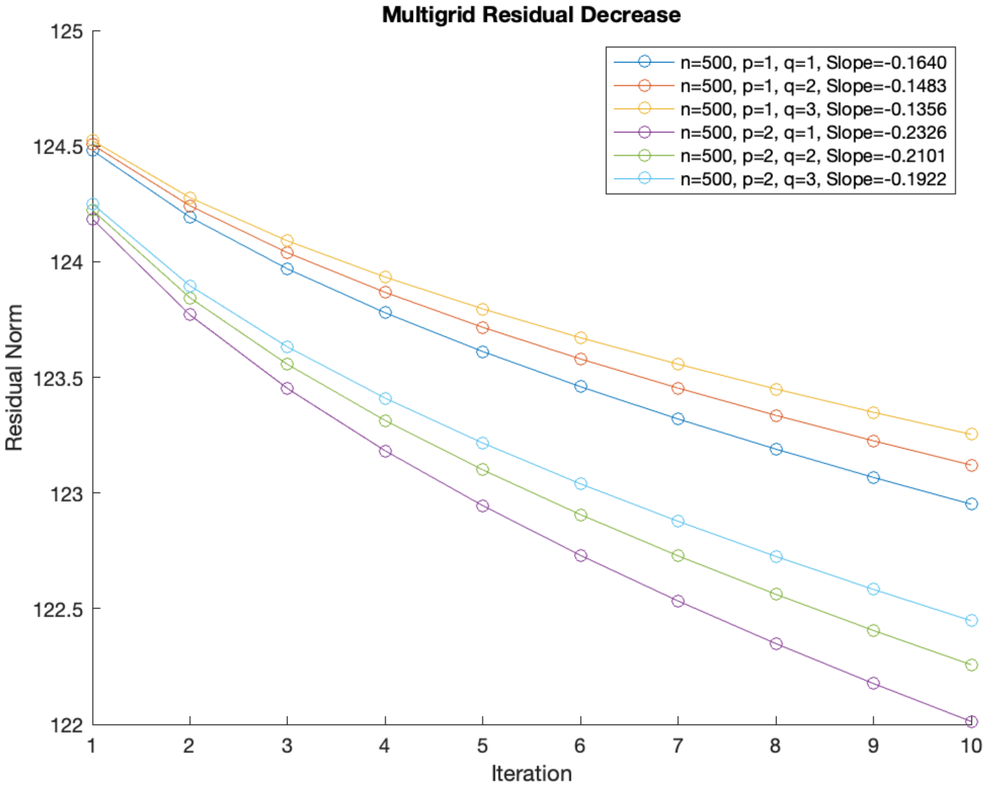
Using different mesh size and keeping p, q the same, we can see that the slopes of a linear fit for them are very close, which implies that the residual goes down by a fixed factor independent of the mesh size.



```
Grid_Size    p     q    Final_Residual    Avg_Reduction_Factor     Slope
_________    _     _    ______________    ____________________    _______

   100       1     1       22.9507               1.0072           -0.1641
   250       1     1       60.4531               1.0028           -0.1638
   500       1     1      122.9515               1.0014           -0.1640
   100       1     2       23.1187               1.0065           -0.1484
   250       1     2       60.6285               1.0025           -0.1474
   500       1     2      123.1205               1.0012           -0.1483
   100       1     3       23.2520               1.0059           -0.1356
   250       1     3       60.7724               1.0023           -0.1335
   500       1     3      123.2533               1.0011           -0.1356
   100       2     1       22.0110               1.0105           -0.2327
   250       2     1       59.5145               1.0040           -0.2324
   500       2     1      122.0124               1.0020           -0.2326
   100       2     2       22.2553               1.0095           -0.2101
   250       2     2       59.7682               1.0036           -0.2091
   500       2     2      122.2574               1.0018           -0.2101
   100       2     3       22.4495               1.0086           -0.1916
   250       2     3       59.9719               1.0033           -0.1897
   500       2     3      122.4476               1.0016           -0.1922
```

For recommendations for p and q: we can see that increasing p from 1 to 2 results in lower final residuals and a greater negative slope, which means that increasing p from 1 to 2 helps to reduce the error more effectively. However, as q increases from 1 to 3, the residuals seems to be higher though the difference is relatively small. It seems like q = 1 would be a reasonable choice, and whether we want to choose p = 2 depends on how much additional work we are willing to pay for the trade-off. We know that taking p = 2 instead of p = 1 will roughly double the work, but it also offers better performance in terms of convergence. If we are using a big fine mesh as quickly as possible, maybe we can just take p = 1 even though the result will not be as good as if we are using p = 2.


Multigrid Residual Decrease
Residual Norm
Iteration
n=500, p=1, q=1, Slope=-0.1640
n=500, p=1, q=2, Slope=-0.1483
n=500, p=1, q=3, Slope=-0.1356
n=500, p=2, q=1, Slope=-0.2326
n=500, p=2, q=2, Slope=-0.2101
n=500, p=2, q=3, Slope=-0.1922