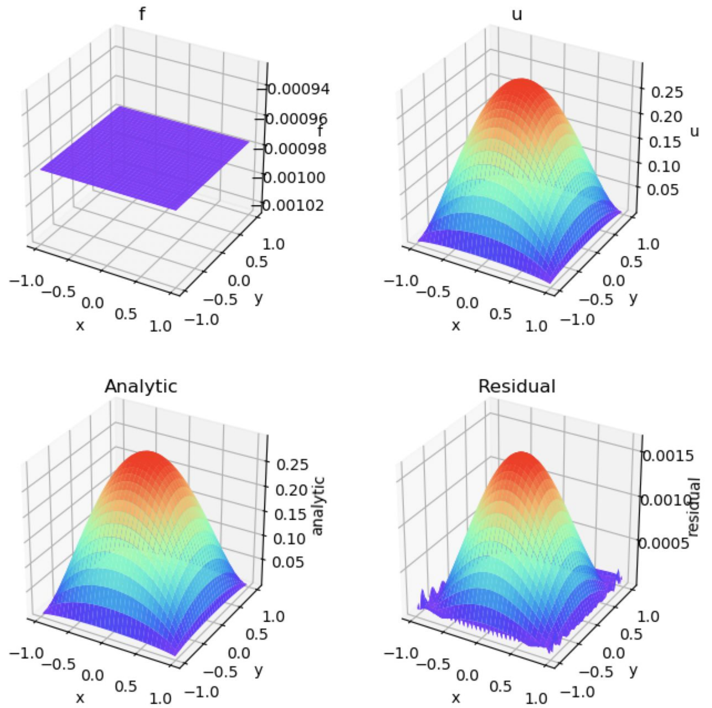
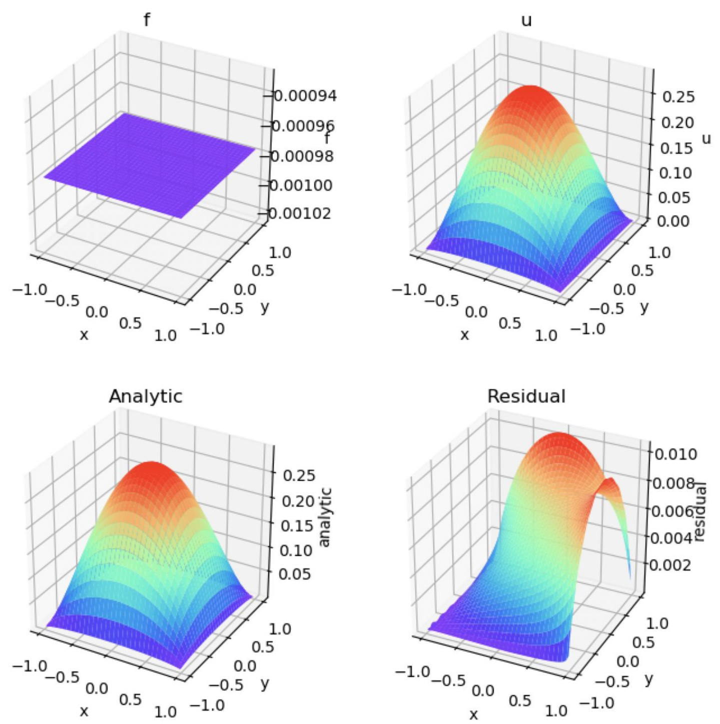
M2(a).



1.
2. Iterations: 4363 Residual: 9.998061251011849e-07
3. The minus sign in front of the RHS of the vector f comes from when you take the steady state solution as t goes to infinity of the diffusion equation and solve for laplacian of u. This reduces down to -f, so that is why the vector is negative.
4. The jacobi method converges when the matrix A is diagonally dominant. If the matrix A is not diagonally dominant, then the eigenvalues may be too large (>1) to converge.

M2(b)

1. In Code



2.
3. In Code, Plots above
4. Iterations: 2428 Residual 9.998811326140166e-07
5.
    a. The iteration matrix for Jacobi is:
        i. -inv(D)*(L+U)
    b. The iteration matrix for Gauss-Seidel is:
        i. -inv(L+D)(U)
    c. By comparing these two matrices we find that the eigenvalues of the Jacobi iteration matrix are the square root of the eigenvalues of the Gauss-Seidel iteration matrix.
    d. This means that if the Jacobi method converges, so will the Gauss-Seidel method and vice versa. It also means that the Gauss-Seidel method will converge approximately twice as fast as the Jacobi method when the methods converge.

MT2(c)

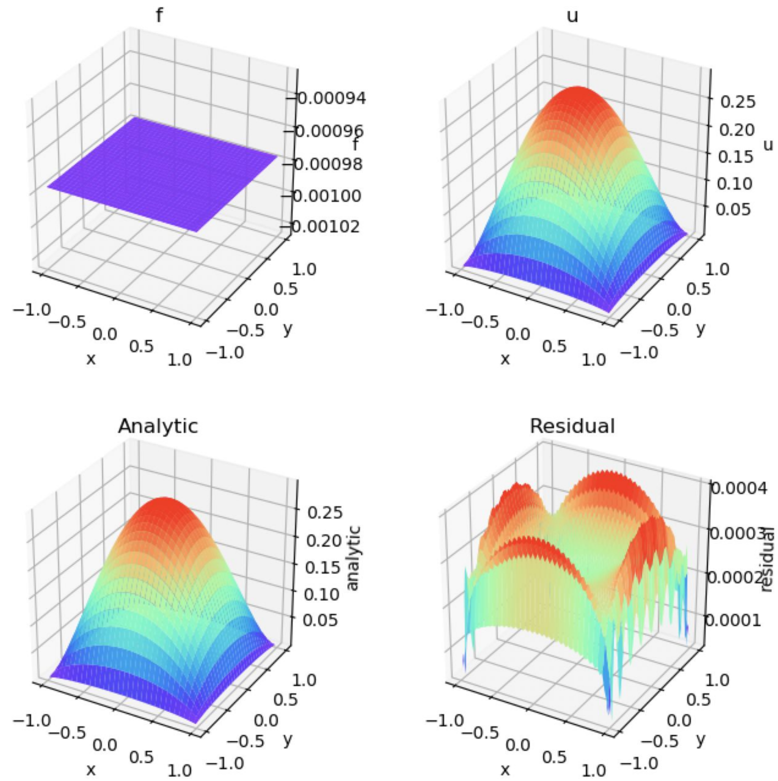
1. Since the relaxation methods possess the smoothing property, small-scale perturbations decay faster than large-scale perturbations. As the wavenumber increases, the wavelengths decrease and the eigenvalues of the system decrease. Since the eigenvalues of the system decrease, the convergence decreases.
    a. The eigenvalues can be found by the equation : cos((1/2)*π*k*(1/n))^2

   b. The convergence of the relaxation method is determined by the eigenvalues
   c. Relaxation methods use neighboring points in order to determine the new value of a point, so if neighboring points are constant it will be extremely slow to converge, but if neighboring points are highly variant (oscillatory), it will be very quick to converge.
2. When you create coarser grids, the mode becomes more oscillatory. This is helpful since we know that the more oscillatory it is, the faster it will converge.
3.
   a. The associated error with the initial guess v_0 is **u**-**v_0**. The associated residual **f**- *A****v_0**. An iteration of the stationary, linear method is **v_1** ← **v_0** + inv(*B*)***r_0**
   b. The solution to the equation *A***e** = **f** - *A***v_0** is **u** - **v_0.** The associated error is (**u** - **v_0**) - **e_0** = **u** - **v_0**. The associated residual is **r_0** - *A***e_0** = **r_0.** Since **e_0** = 0 and **e_1** ← **e_0** + inv(*B*)(**r_0** - *A***e_0**), then **e_1** ← **e_0** + inv(*B*)r**_0** .
   c. The solution to the second system is shifted by **v_0**. However, the associated error and residual equations are equivalent.

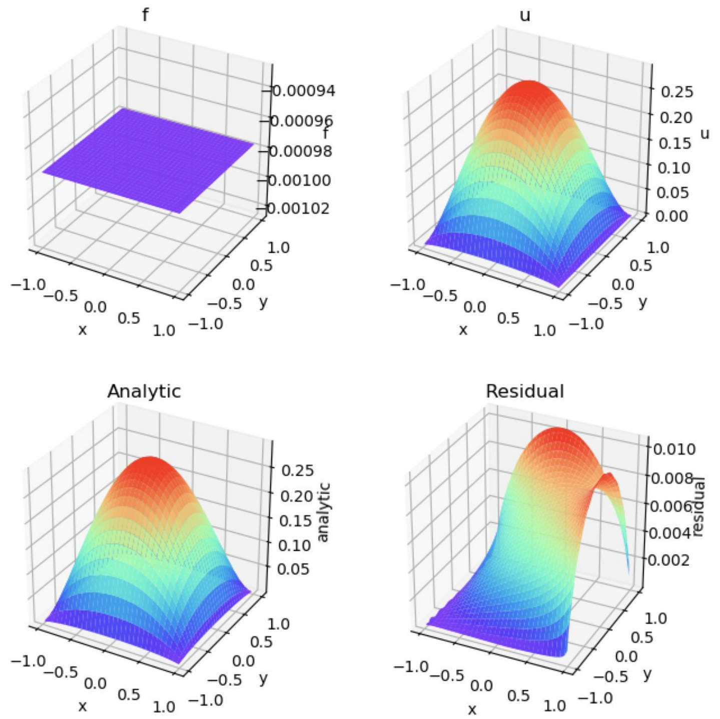
MT2(d)

1.
   a. mg_restrict(u) restricts u to half the grid size
   b. mg_prolong(u,fBNC) interpolates u to twice the grid size
   c. mg_residual(u,f,fBNC) calculates the residual
   d. mg_vcycle(f,fBNC,npre,npst,level,tol,**kwargs) recursively solves the multigrid
      i. Precondition
      ii. Call residual
      iii. Then restrict rhs
      iv. Then restrict vh
      v. Recurse
      vi. Then prolong e2h
      vii. Add correction
      viii. Smooth noise
   e. multigrid(f,fBNC,tol, **kwargs) repeatedly calls mg_vcycle once per iteration to drive the process
2. In the code
3. In the code
4. Already provided
5. Already provided

MT2(e)

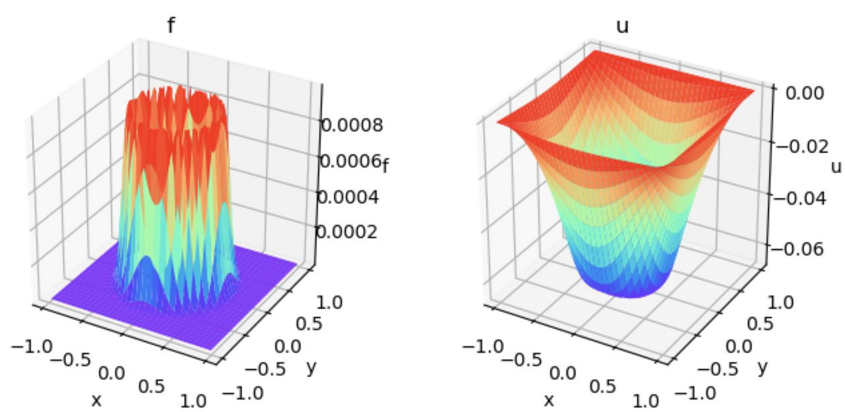


1.
   a. Takes more than 20 iterations (but I am using Jacobi instead of Gauss-Seidel for pre-conditioning and prolongation)
      i. Iteration: 141 Residual: 9.721755557839658e-07



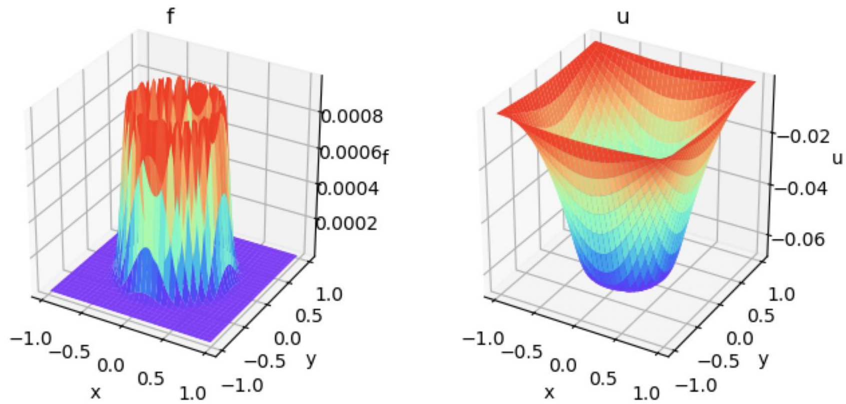
2.

a. GS Iterations: 2428 Residual 9.998811326140166e-07
b. The residuals in the multigrid method are of smaller order of magnitude. The multigrid method is much quicker than my Gauss-Seidel.



3.

a. Iterations: 1850 Residual: 9.978312334166997e-07



4.

a. Iteration: 138 Residual: 9.714062049877429e-07
b. The multigrid is much shorter