Karin Metzgar
ASTE 546
HW 13

Update the initial function

```
double f0 = 1.0/sqrt(2*pi)*exp(-v*v/2.0);
            //Number 1 initialize new funciton
            f[i][j] = f0 * (1.0 + cos((2 * pi * x) / world.L));
            f[i][j] = f0;
```

Create a function that makes the domain periodic – loop over all j values and zip the i's together by making them both equal the average between them

```
void makePeriodic(double **g, int ni, int nj){

    for (int j=0;j<nj;j++)
    {
        double average = 0.5 * (g[0][j] + g[ni-1][j]);
        g[0][j] = average;
        g[ni-1][j] = average;
    }
}
```

Call the function after //compute f(n+1) loop

Make the gather function work for period domain

```
//linear interpolation: a higher order scheme needed!
    double interp(double **f, double x, double v)
    {
        //this version returns zero if out of bounds
        double fi = (x-0)/dx;
        double fj = (v-(-v_max))/dv;

        //no longer out of bounds - adjusting to "wrap"
        //if (fi<0 || fi>ni-1) return 0;
        //if (fj<0 || fj>nj-1) return 0;

        //get the remainer so domain doesnt keep increasing
        fi = fmod(fi + ni, ni);
        fj = fmod(fj + nj, nj);

        //get rid of the integer part = di, dj
        int i = (int)fi;
        int j = (int)fj;
        double di = fi-i;
        double dj = fj-j;
```

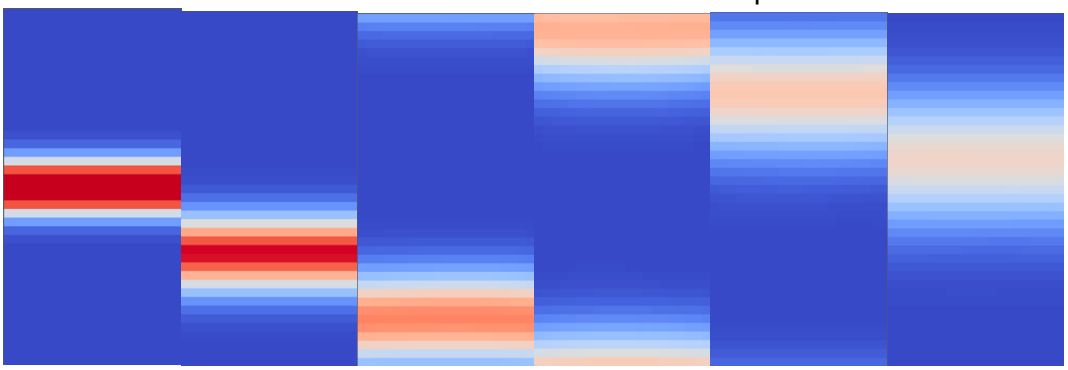
```
        //points to the "right", % allows it to wrap
        int ip1 = (i+1) % ni;
        int jp1 = (j+1) % nj;

        //bottom left point
        double val = (1-di)*(1-dj)*f[i][j];
        //if (i<ni-1) val+=(di)*(1-dj)*f[i+1][j];
        //if (j<nj-1) val+=(1-di)*(dj)*f[i][j+1];
        //if (i<ni-1 && j<nj-1) val+=(di)*(dj)*f[i+1][j+1];
        //bottom right point
        val += (di)*(1-dj)*f[ip1][j];
        //top left point
        val += (1-di)*(dj)*f[i][jp1];
        //top right point
        val += (di)*(dj)*f[ip1][jp1];


        return val;

    }
```

This is the animation for f over time .. where left and right is y (v) ... doesn't look like v = 0 is stationary? I can't get Paraview to visualize the results from the example vlasov.cpp or my code in 3D but this looks much different than the example 2D simulation ran.



Replace solvePoissonsEquation with GaussSeidel method:

```
void gaussSeidel(double* x, double* ne, double dx, int ni, int max = 1000, double
tolerance = 1e-6) {
    double dx2 = dx * dx;
    bool converged = false;

    for (int iter = 0; iter < max; ++iter) {
```

```
        converged = true;
        for (int i = 0; i < ni; ++i) {
            // apply periodic boundary conditions from the main code I added
            int ip1 = (i + 1) % ni;
            int im1 = (i - 1 + ni) % ni;

            // Gauss-Seidel update
            double new_x = 0.5 * (x[im1] + x[ip1] - dx2 * (1-ne[i]));

            // Check for convergence
            if (fabs(new_x - x[i]) > tolerance) {
                converged = false;
            }

            x[i] = new_x;
        }

        if (converged) {
            break;
        }
    }
}
```

The results I'm getting look pretty weird to me but .. not sure.

Final results are in resutls1 folder.