Numerical Solver of 2D Infinite Well

This script is a numerical Solver for the 2D infinite well scenario utilising finite differences of the second order derivative.

Preparing Values

m is mass, where here it is set to that of the electron rest mass in $\mathrm{eV}\cdot s^2\cdot m^{-2}$.
A is the width of the infinite well in m^2.
a is the width of the segments of the infinite well for the finite differences approximation, in the same units.

```
m = 0.511 * 10^6 / (9 * 10^16);
Ax = 0.53 * 10^-10;
Ay = 0.73 * 10^-10;
```

nx is the number of points segmenting the domain of the infinite well in the x direction, and ny in the y direction.
ie. n = sidelength - 2

```
nx = 4;
ny = 10;
```

```
ax = Ax / (nx + 1);
ay = Ay / (ny + 1);
```

node is the *n*th harmonic to be displayed by the graphing solution below.

```
node = 5;
```

Preparing Eigensystem

```
ij = Flatten[Table[{i, j}, {j, ny}, {i, nx}], 1];
```

```
getMatrixElement[k1_, k2_, nx_, ny_] :=
 If[k1 == k2, 4,
  If[k1 == k2 - 1 || k1 == k2 + 1 || k1 == k2 + nx || k1 == k2 - nx, -1, 0]]
```

Here we construct the matrix resulting from applying the finite differences approximation to the TISE.

```
finiteDifferencesMatrix = Table[
   Table [
    getMatrixElement[j, i, nx, ny]
    , {j, nx * ny}]
   , {i, nx * ny}];
```

Solving Eigenproblem and Plotting Results

Solve the eigenproblem and obtain the resulting eigensystem.

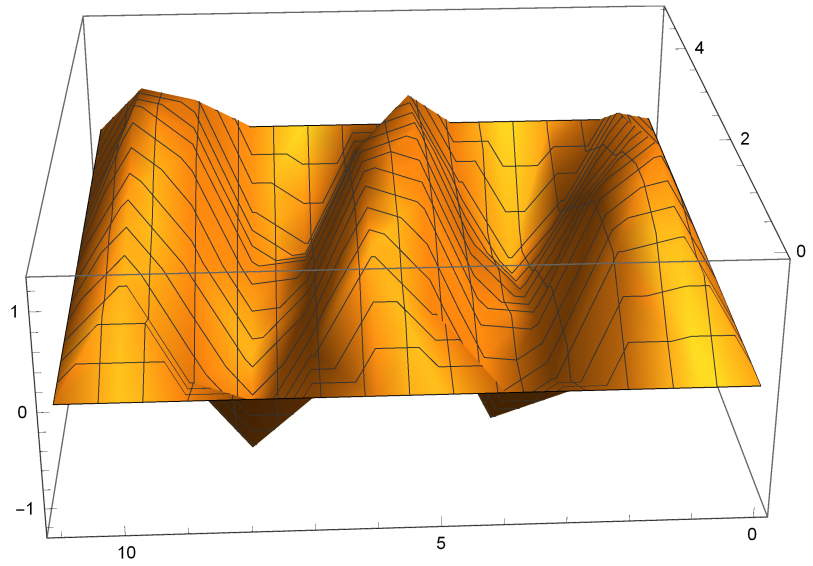
```
eigensystem = Eigensystem[finiteDifferencesMatrix];
```

This shows the first harmonic of the wave function for a particle in the constructed infinite well, and its respective energy.

```
plot = Table[Join[ij[[k]], {eigensystem[[2, -node, k]]}], {k, nx * ny}];
For[x = 0, x < (nx + 2), x++, AppendTo[plot, {x, 0, 0}]];
For[x = 0, x < (nx + 2), x++, AppendTo[plot, {x, ny + 1, 0}]];
For[y = 1, y < (ny + 1), y++, AppendTo[plot, {0, y, 0}]];
For[y = 1, y < (ny + 1), y++, AppendTo[plot, {nx + 1, y, 0}]];
```

Plot results!

```
ListPlot3D[plot]
```



Plot of the square of the wavefunction:

```
plot2 = Table[Join[ij[[k]], {eigensystem[[2, -node, k]]^2}], {k, nx * ny}];
For[x = 0, x < (nx + 2), x++, AppendTo[plot2, {x, 0, 0}]];
For[x = 0, x < (nx + 2), x++, AppendTo[plot2, {x, ny + 1, 0}]];
For[y = 1, y < (ny + 1), y++, AppendTo[plot2, {0, y, 0}]];
For[y = 1, y < (ny + 1), y++, AppendTo[plot2, {nx + 1, y, 0}]];
```

```
ListPlot3D[plot2]
```