Rectangular Drumhead

Completed and Analyzed in class, March 25, 2025

This is the fourteenth notebook for you to complete. It is our first notebook that is has a two-dimensional network of masses. We'll make those two dimensions be the *x* and *y* directions. The two-dimensional network of masses will oscillate vertically (in the *z* direction).

Initial Conditions

Set up the duration, **steps**, and **deltaT**:

```
In[ ]:= tInitial = 0.0;
tFinal = 10.0;
steps = 5000;
deltaT = (tFinal - tInitial) / steps;
```

Set up the size of the grid (enough masses so that the grid looks like a drumhead, but not so many that we tax our computers):

```
nx = 18; (* There is actually going to be 19, but both x edges will be fixed. *)
(* So the net number that are actually moving will be 17 in the x-direction. *)
ny = 24; (* There is actually going to be 25, but both y edges will be fixed. *)
(* So the net number that are actually moving will be 23 in the y-direction. *)
(* 17 * 23 means that the computer is simulating a grid of 391 masses. *)
(* It is doing this for 5000 time steps so in all your computer is having to *)
(* compute and render about 2000000 particle positions. *)
```

We are going to make initial conditions that are a product of sine functions. What sine function specifically is specified by the modes.

```
In[ ]:= modex = 2;
modey = 3;
maxz = 1.0;
initialzs =
  Table[maxz Sin[Pi modex (j - 1) / nx] Sin[Pi modey (k - 1) / ny], {j, nx + 1}, {k, ny + 1}];
initialvs = Table[0, {j, nx + 1}, {k, ny + 1}];
initialConditions = {tInitial, initialzs, initialvs};
```

Formulas for the Accelerations — Theory

The acceleration formula

$$a_{j,k} = v_0^2\left(z_{j,k+1} + z_{j,k-1} + z_{j+1,k} + z_{j-1,k} - 4\,z_{j,k}\right)$$

is valid except for the edges, and we have to handle those separately.

Fixed Edges

A drumhead is normally fixed at the edges, and we are going to deal with the edges by just freezing the edge masses to have $z = 0$. So $z_{1,k} = 0$, $z_{n_x+1,k} = 0$, $z_{j,1} = 0$, and $z_{j,n_y+1} = 0$.

Conceptually, you can think of the index j as running from 0 to n_x and the index k as running from from 0 to n_y, but that goes against the grain of the way Mathematica indexes its arrays, so we are going to have to be super-careful about off-by-one errors. The index j will run from 1 to $n_x + 1$ and the index k will run from 1 to $n_y + 1$.

You can see that the necessary care was already taken in the initialConditions above.

Implementing the Accelerations

```
v0 = 4 Pi;

a[j_, k_, allzs_] := v0^2 If[j == 1 || chattanooga,
   0, (* no acceleration at the edges *)
   allzs[[j, k + 1]] + choo
  ]
```

Second-Order Runge-Kutta — Implementation

```
In[ ]:= rungeKutta2[cc_] := (
    curTime = cc[[1]];
    curzs = cc[[2]];
    curvs = cc[[3]];
    newTime = curTime + deltaT;
    zsStar = curzs + curvs deltaT / 2;
    as = Table[a[j, k, zsStar], {j, 1, nx + 1}, {k, 1, ny + 1}];
    newvs = curvs + as deltaT;
    newzs = curzs + (curvs + newvs) deltaT / 2;
    {newTime, newzs, newvs}
   )

rk2Results = NestList[rungeKutta2, initialConditions, steps];

rk2ResultsTransposed = Transpose[rk2Results];
zs = rk2ResultsTransposed[[2]];
```

3D Graphics

We need a graphics implementation with $(n_x + 1)(n_y + 1)$ masses. We'll space the masses equally across the x and y axes of the cuboid and draw grid lines connecting them.

```
halfHeight = 1;
halfDepth = 4;
halfWidth = 3;
xspacing = 2 halfWidth / nx;
yspacing = 2 halfDepth / ny;
cuboid = {FaceForm[{Blue, Opacity[0.04]}], Cuboid[
    {-halfWidth, -halfDepth, -halfHeight}, {halfWidth, halfDepth, halfHeight}]};
drumheadGraphic[zs_] := Graphics3D[Flatten[{
      {cuboid},
      Table[
       Point[{-halfWidth + (j - 1) xspacing, -halfDepth + (k - 1) yspacing, zs[[j, k]]}],
       {j, nx + 1}, {k, ny + 1}
      ],
      Table[
       Line[{{-halfWidth + (j - 1) xspacing, -halfDepth + (k - 1) yspacing, zs[[j, k]]},
         {-halfWidth + (j - 1) xspacing, -halfDepth + k yspacing, zs[[j, k + 1]]}}],
       {j, nx + 1}, {k, ny}
      ],
      (* All the points and all the grid lines that go in the y-direction *)
      (* are already done. Add the grid lines that go in the x-direction. *)
      choo
     }, 1]];
drumheadGraphic[initialzs]
```

Animating the 3D Graphics

The default duration of the animation is the duration of our simulation:

```
In[ ]:= Animate[drumheadGraphic[zs[[step]]],
  {step, 0, steps, 1}, DefaultDuration → tFinal - tInitial]
```