Lemmings on a plank

Spencer and Justin
June 12, 2017

Dad said:

100 lemmings are positioned at 1-meter intervals along a narrow, 101-meter-long floating plank. (This is a 1-dimensional problem. Picture the plank as oriented left-right. The two outermost lemmings are 1 meter from the ends. Consider the lemmings to have zero length.) Each lemming is facing either left or right. At t=0, each lemming starts moving forward at 1 m/s. When two lemmings collide, each instantly reverses his direction of motion. Lemmings that reach the ends fall into the water and are out of play.

If you can dictate the initial orientation of each of the lemmings, i.e., choose each lemming's initial direction of motion, what is the longest time the game can last before all the lemmings are in the water?

Let's simulate this hybrid dynamical system to get a feel for it.

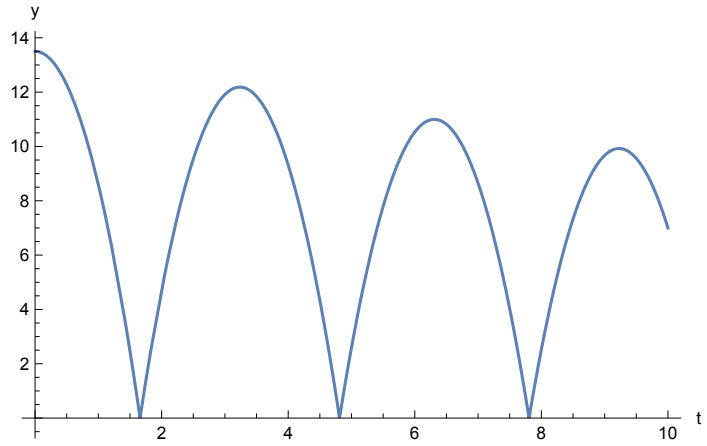
In[1]:=
```
Clear["Global`*"]
```

NDSolve[] bouncing ball example

In[2]:=
```
ball = NDSolve[{y''[t] == -9.81, y[0] == 13.5, y'[0] == 0,
    WhenEvent[y[t] == 0, y'[t] -> -0.95 y'[t]]}, y, {t, 0, 10}];
```

In[3]:=
```
Plot[y[t] /. ball, {t, 0, 10}, AxesLabel → {"t", "y"}]
```

Out[3]=



With 2 lemmings

In[4]:=
```
soln = NDSolve[{
    x[0] == 1,
    x'[0] == 1,
    y[0] == 2,
    y'[0] == -1,
    x''[t] == 0,
    y''[t] == 0,
    WhenEvent[x[t] == y[t], {y'[t] -> -y'[t], x'[t] -> -x'[t]}],
    WhenEvent[{x[t] < 0, y[t] > 3}, "StopIntegration"]
  }, {x, y}, {t, 0, 10}]
```

Out[4]=
```
{{x → InterpolatingFunction[  ],
  y → InterpolatingFunction[  ]}}
```

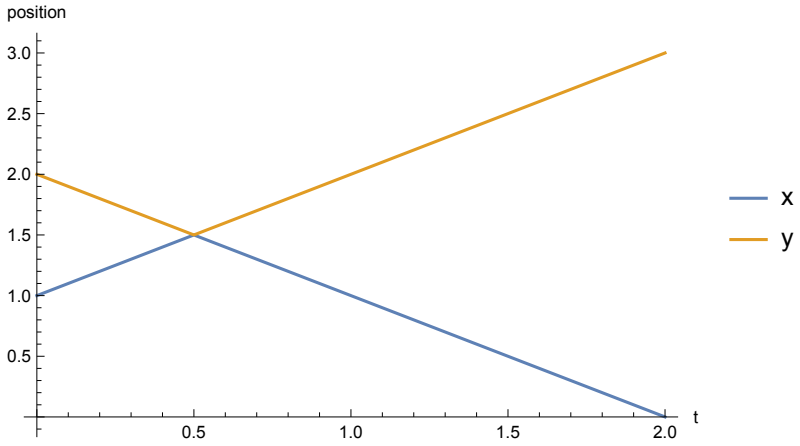

Hack into the InterpolatingFunction to find its max domain:

In[5]:=
```
max = soln[[1, 1, 2, 1, 1, 2]]
```

Out[5]=
```
2.
```

In[6]:=
```
Plot[Evaluate[{x[t], y[t]} /. soln], {t, 0, max},
 AxesLabel → {"t", "position"}, PlotLegends → {"x", "y"}]
```

Out[6]=



With *n* lemmings

In[7]:=
```
n = 10;
vars = Array[x, n]
```

Out[8]=
```
{x[1], x[2], x[3], x[4], x[5], x[6], x[7], x[8], x[9], x[10]}
```

In[9]:=
```
eqns = {};
For[i = 1, i ≤ n, i++,
  AppendTo[eqns, {
    x[i][0] == i,
    x[i]'[0] == If[i ≤ n / 2, 1, -1],
    x[i]''[t] == 0,
    If[i < 10, WhenEvent @@ Evaluate[{
        x[i][t] == x[i + 1][t],
        {x[i]'[t] → -x[i]'[t], x[i + 1]'[t] → -x[i + 1]'[t]}
       }], Nothing]
   }]];
```

If there are no lemmings on the plank, stop.

In[11]:=
```
allgone = And @@ Table[x[i][t] < 0 || x[i][t] > n + 1, {i, n}];
```

In[12]:=
```
AppendTo[eqns, WhenEvent[Evaluate@allgone, "StopIntegration"]];
```

In[13]:=
```
eqns = Flatten@eqns;
Column[eqns]
```

Out[14]=
```
x[1][0] == 1
x[1]'[0] == 1
x[1]''[t] == 0
WhenEvent[x[1][t] == x[2][t], {x[1]'[t] → -x[1]'[t], x[2]'[t] → -x[2]'[t]}]
x[2][0] == 2
x[2]'[0] == 1
x[2]''[t] == 0
WhenEvent[x[2][t] == x[3][t], {x[2]'[t] → -x[2]'[t], x[3]'[t] → -x[3]'[t]}]
x[3][0] == 3
x[3]'[0] == 1
x[3]''[t] == 0
WhenEvent[x[3][t] == x[4][t], {x[3]'[t] → -x[3]'[t], x[4]'[t] → -x[4]'[t]}]
x[4][0] == 4
x[4]'[0] == 1
x[4]''[t] == 0
WhenEvent[x[4][t] == x[5][t], {x[4]'[t] → -x[4]'[t], x[5]'[t] → -x[5]'[t]}]
x[5][0] == 5
x[5]'[0] == 1
x[5]''[t] == 0
WhenEvent[x[5][t] == x[6][t], {x[5]'[t] → -x[5]'[t], x[6]'[t] → -x[6]'[t]}]
x[6][0] == 6
x[6]'[0] == -1
x[6]''[t] == 0
WhenEvent[x[6][t] == x[7][t], {x[6]'[t] → -x[6]'[t], x[7]'[t] → -x[7]'[t]}]
x[7][0] == 7
x[7]'[0] == -1
x[7]''[t] == 0
WhenEvent[x[7][t] == x[8][t], {x[7]'[t] → -x[7]'[t], x[8]'[t] → -x[8]'[t]}]
x[8][0] == 8
x[8]'[0] == -1
x[8]''[t] == 0
WhenEvent[x[8][t] == x[9][t], {x[8]'[t] → -x[8]'[t], x[9]'[t] → -x[9]'[t]}]
x[9][0] == 9
x[9]'[0] == -1
x[9]''[t] == 0
WhenEvent[x[9][t] == x[10][t], {x[9]'[t] → -x[9]'[t], x[10]'[t] → -x[10]'[t]}]
x[10][0] == 10
x[10]'[0] == -1
x[10]''[t] == 0
WhenEvent[(x[1][t] < 0 || x[1][t] > 11) && (x[2][t] < 0 || x[2][t] > 11) &&
  (x[3][t] < 0 || x[3][t] > 11) && (x[4][t] < 0 || x[4][t] > 11) &&
  (x[5][t] < 0 || x[5][t] > 11) && (x[6][t] < 0 || x[6][t] > 11) &&
  (x[7][t] < 0 || x[7][t] > 11) && (x[8][t] < 0 || x[8][t] > 11) &&
  (x[9][t] < 0 || x[9][t] > 11) && (x[10][t] < 0 || x[10][t] > 11), StopIntegration]
```

In[15]:=
```
lem = NDSolveValue[Flatten@eqns, vars, {t, 0, 100}]
```

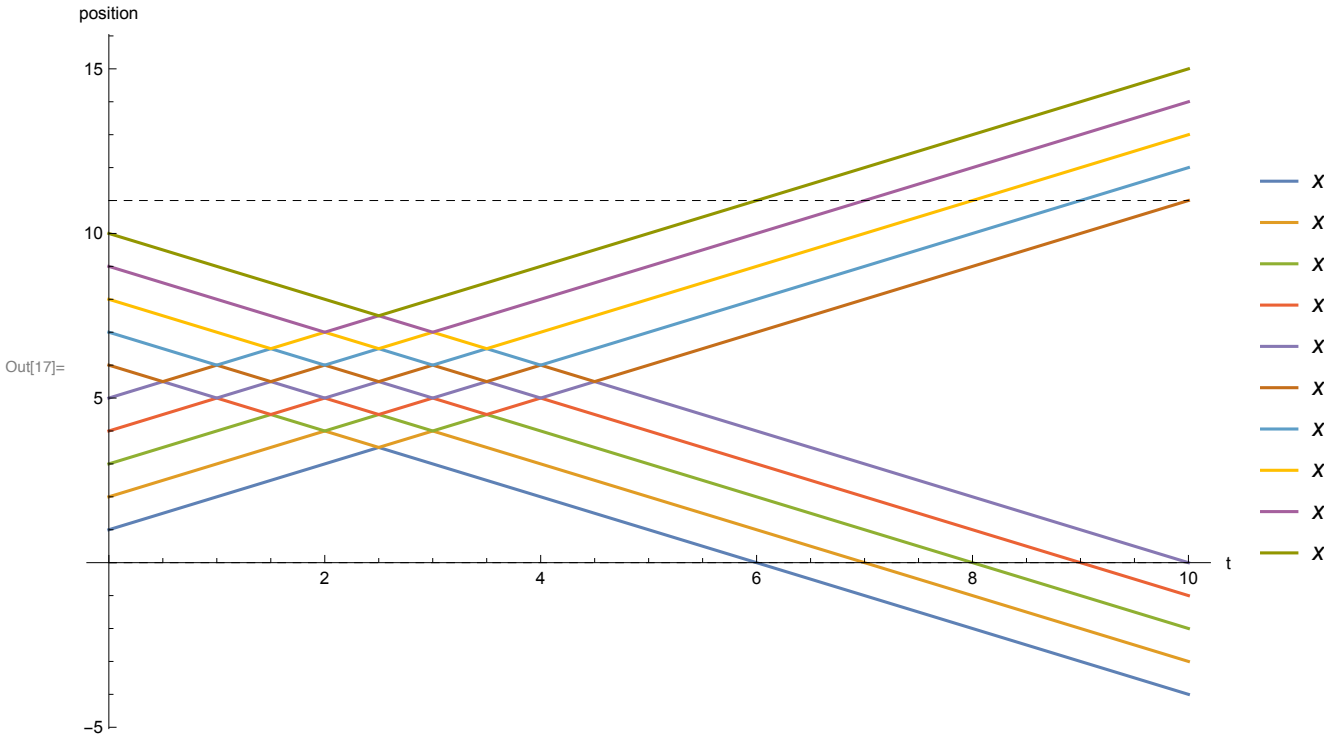
Out[15]=
```
{InterpolatingFunction[  ],
 InterpolatingFunction[  ],
 InterpolatingFunction[  ],
 InterpolatingFunction[  ],
 InterpolatingFunction[  ],
 InterpolatingFunction[  ],
 InterpolatingFunction[  ],
 InterpolatingFunction[  ],
 InterpolatingFunction[  ],
 InterpolatingFunction[  ]}
```


In[16]:=
```
tmax = lem[[1, 1, 1, 2]]
```

Out[16]=
```
10.
```

In[17]:=
```
Plot[Evaluate[Through[lem[t]]], {t, 0, tmax},
 AxesLabel → {"t", "position"}, PlotLegends → vars, ImageSize → 600,
 Epilog → {Dashed, Line[{{0, 0}, {tmax, 0}}], Line[{{0, n + 1}, {tmax, n + 1}}]}]
```

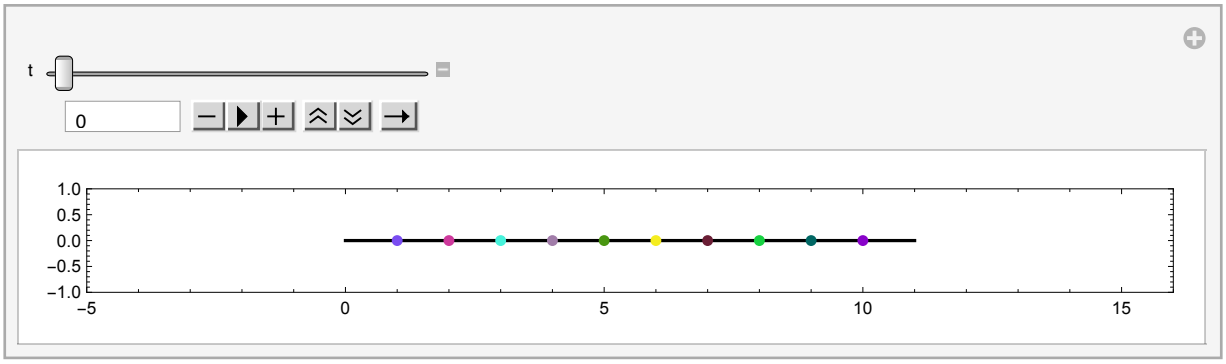
Out[17]=



In[18]:=
```
colors = RandomColor@Length@lem;
```

In[19]:=
```
Manipulate[Graphics[{
    Thickness[.003],
    Line[{{0, 0}, {11, 0}}],
    PointSize[.01],
    {colors, Point[{#, 0}] & /@ Through[lem[t]]}ᵀ
  },
  Axes → False,
  Frame → True,
  PlotRange → {{-5, 16}, {-1, 1}},
  ImageSize → Full],
 {t, 0, tmax, Appearance → "Open"}]
```

Out[19]=



Spence pointed out that two lemmings bouncing off each other is actually the same as the lemmings just passing through each other. He pointed out that if the lemmings had nonzero width, the game would end sooner because they'd essentially teleport through teach other as they passed through.