```
Manipulate[
 (*Nasser M. Abbasi, july 7, 2014*)
 tick;

 Module[{j, x, t, graph, h, i, fun, len = 1, end = -1},
  h = len / (nPoints - 1);
  t = h / c;
  fun = Which[f == 1, f1, f == 2, f2];

  If[dataChanged == True,
   dataChanged = False;
   n = 0];

  Which[
   n == 0,
   uNow = Table[0, {i, nPoints}];
   uNext = uLast = uNow;
   grid = Table[j h, {j, 0, nPoints - 1}];
   uNow = fun[#] & /@ grid,

   n == 1,
   uNext[[1]] = 0;
   uNext[[2 ;; -2]] = (1 / 2) (uNow[[1 ;; end - 2]] + uNow[[3 ;; end]]);
   uNext[[end]] = 0;
   uLast = uNow;
   uNow = uNext,

   n > 1,
   uNext[[1]] = 0;
   uNext[[2 ;; (end - 1)]] =
    uNow[[1 ;; (end - 2)]] + uNow[[3 ;; end]] - uLast[[2 ;; (end - 1)]];
   uNext[[end]] = 0;
   uLast = uNow;
   uNow = uNext
  ];

  graph = Grid[{
      {Row[{"time", Spacer[5], padIt2[N[n * t], {6, 5}],
          Spacer[5], "step number", Spacer[2], padIt2[n, 6]}]},
      (*{ScientificForm[N@(n-1)*t,4]},*)
      {ListLinePlot[Transpose[{grid, uNow}], PlotRange → {{-.1, 1.1}, {-1, 1}},
        ImageSize → 400, GridLines → Automatic, GridLinesStyle → LightGray], SpanFromLeft}
     }, Alignment → Center
   ];

  Which[state == "RUN" || state == "STEP",
   n++;
   If[state == "RUN",
    tick = Not[tick]
   ]
  ];
 graph
],

Grid[{
  {
   Grid[{
     { Button[Text@Style["run", 12],
       {state = "RUN"; tick = Not[tick]}, ImageSize → {60, 40}],
      Button[Text@Style["step", 12], {state = "STEP"; tick = Not[tick]},
       ImageSize → {60, 40}],
      Button[Text@Style["stop", 12], {state = "STOP"; tick = Not[tick]},
       ImageSize → {60, 40}],
      Button[Text@Style["reset", 12], {state = "RESET"; nPoints = 300;
        dataChanged = True; n = 0; c = 351; f = 2; tick = Not[tick]}, ImageSize → {60, 40}]}
    }, Spacings → {.5, 0}, Frame → True, FrameStyle → Gray
   ], SpanFromLeft
  },
  {"number of x-points",
   Manipulator[Dynamic[nPoints, {nPoints = #; dataChanged = True; tick = Not[tick]} &],
    {5, 1000, 1}, ImageSize → Tiny], Dynamic[padIt2[nPoints, 4]], SpanFromLeft},
  {"wave speed", Manipulator[Dynamic[c, {c = #; dataChanged = True; tick = Not[tick]} &],
    {1, 1000, 1}, ImageSize → Tiny], Dynamic[padIt2[c, 4]], SpanFromLeft},
  {"select initial data",
   SetterBar[Dynamic[f, {f = #; dataChanged = True; tick = Not[tick]} &], {1, 2}
   ], SpanFromLeft
  }
 }, Alignment → Center, Spacings → {0.5, .2}],

{{f, 2}, None},
{{state, "STOP"}, None},
{{n, 0}, None},
{{c, 351}, None},
{{dataChanged, True}, None},
{{nPoints, 300}, None},
{{tick, False}, None},
{{uNow, {}}, None},
{{uNext, {}}, None},
{{uLast, {}}, None},
{{grid, {}}, None},
TrackedSymbols :→ {tick},
ControlPlacement → Left, Alignment → Center, ImageMargins → 0, FrameMargins → 0,
Initialization :→
 (
  f1[x_] := Piecewise[{{Sin[2 Pi (x - 0.2) / 0.4], 0.2 <= x ≤ 0.6}, {0, True}}];
  f2[x_] := Piecewise[{{x, 0 <= x ≤ 0.5}, {1 - x, True}}];
  len = 1;
  integerStrictPositive = (IntegerQ[#] && # > 0 &);
  integerPositive = (IntegerQ[#] && # ≥ 0 &);
  numericStrictPositive = (Element[#, Reals] && # > 0 &);
  numericPositive = (Element[#, Reals] && # ≥ 0 &);
  numericStrictNegative = (Element[#, Reals] && # < 0 &);
```

```
   numericNegative = (Element[#, Reals] && # ≤ 0 &);
   bool = (Element[#, Booleans] &);
   numeric = (Element[#, Reals] &);
   integer = (Element[#, Integers] &);
   (*--------------------------------------------*)
   padIt1[v_?numeric, f_List] := AccountingForm[v,
     f, NumberSigns → {"-", "+"}, NumberPadding → {"0", "0"}, SignPadding → True];
   (*--------------------------------------------*)
   padIt1[v_?numeric, f_Integer] := AccountingForm[Chop[v],
     f, NumberSigns → {"-", "+"}, NumberPadding → {"0", "0"}, SignPadding → True];
   (*--------------------------------------------*)
   padIt2[v_?numeric, f_List] := AccountingForm[v,
     f, NumberSigns → {"", ""}, NumberPadding → {"0", "0"}, SignPadding → True];
   (*--------------------------------------------*)
   padIt2[v_?numeric, f_Integer] := AccountingForm[Chop[v],
     f, NumberSigns → {"", ""}, NumberPadding → {"0", "0"}, SignPadding → True]
   (*--------------------------------------------*)
  )

]
```