ArcSine laws Numerical 07-05-16

N. T. Gladd

Initialization: Be sure the files *NTGStylesheet2.nb* and *NTGUtilityFunctions.m* is are in the same directory as that from which this notebook was loaded. Then execute the cell immediately below by mousing left on the cell bar to the right of that cell and then typing “shift” + “enter”. Respond “Yes” in response to the query to evaluate initialization cells.

In[9]:=

```
SetDirectory[NotebookDirectory[]];
(* set directory where source files are located *)
SetOptions[EvaluationNotebook[], (* load the StyleSheet *)
 StyleDefinitions → Get["NTGStylesheet2.nb"]];
Get["NTGUtilityFunctions.m"]; (* Load utilities package *)
```

Purpose

For motivation and background see the notebook *Discrete Random Walks 07-03-16*.

The Mathematica documentation for ArcSineDistribution has some nice examples of the calculation of the three arcsine laws for Brownian motions. I recast those examples in my coding style and embellish a bit.

I return to this notebook and broaden the content to cover arcsine laws as they pertain to Bernoulli trials with skewed payouts. This is a situation that approximately corresponds to my premium selling option campaigns. I use simulation models developed in the context of Kelly criterion analyses — *C:\NT-G\2016\Analysis\Kelly Simulation 04-04-16*.

Brownian motion

First arcsine law: arcsine law of leads

For reasons discussed in *Discrete Random Walks 07-03-16*, I use Brownian motion to model the result from a campaign of trades.

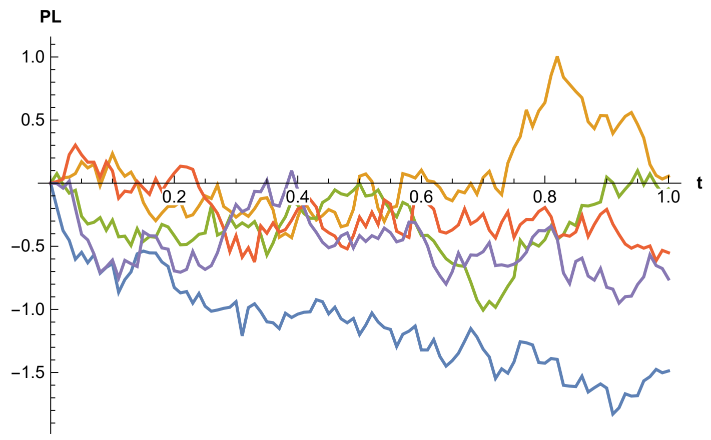
In[11]:=

```
Clear[PLAsBrownianMotion];
PLAsBrownianMotion[n_] := RandomFunction[WienerProcess[], {0, 1, 1/n}]
```

In[13]:=

```
Module[{nSteps = 100, nCases = 5},
 ListLinePlot[Table[PLAsBrownianMotion[nSteps], {nCases}],
  AxesLabel → {Stl["t"], Stl["PL"]}]]
```

Out[13]=



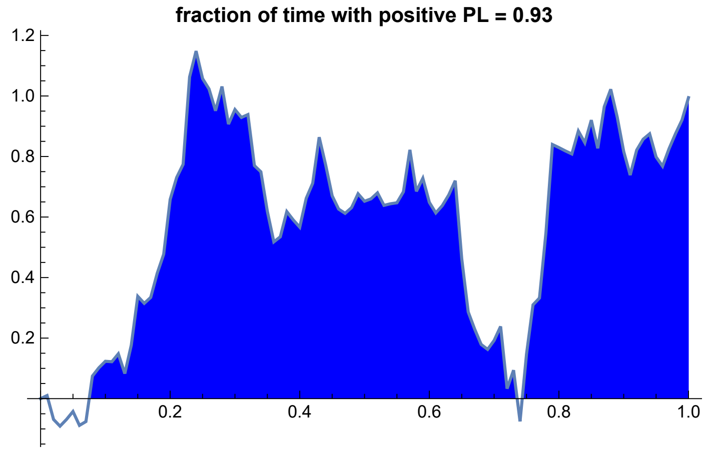
In[3]:=

```
Clear[FractionalTimePositivePL];
FractionalTimePositivePL[n_] :=
 Count[PLAsBrownianMotion[n]["Values"], _?Positive] / n // N
```

In[14]:=

```
Module[{nSteps = 100, data, fractionTimePositive, lab},
 data = PLAsBrownianMotion[nSteps];
 fractionTimePositive = N[Count[data["Values"], _?Positive] / nSteps];
 lab =
  Stl@StringForm["fraction of time with positive PL = ``", fractionTimePositive];
 ListLinePlot[data, Filling → Axis, FillingStyle → {White, Blue}, PlotLabel → lab]]
```

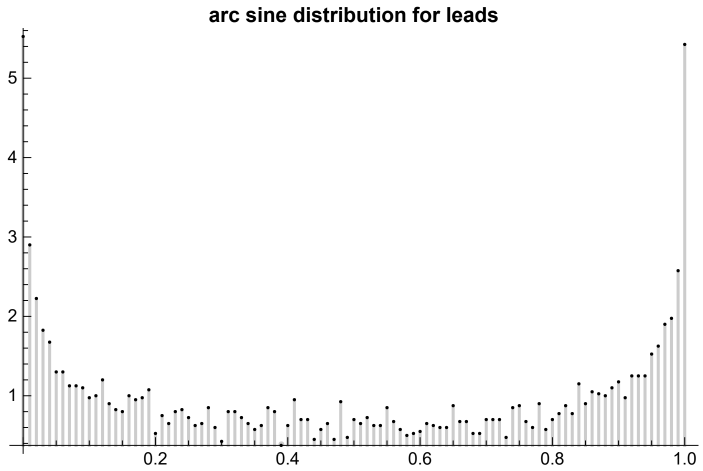
Out[14]=



In[15]:=

```
Module[{nSteps = 100, nCases = 4000,
  nBins = 100, data, ℋ, lab, FractionalTimePositivePL},
 FractionalTimePositivePL[n_] :=
  Count[PLAsBrownianMotion[n]["Values"], _?Positive] / n // N;
 data = Table[FractionalTimePositivePL[nSteps], {nCases}];
 ℋ = HistogramDistribution[data, nBins];
 lab = Stl@StringForm["arc sine distribution for leads"];
 DiscretePlot[PDF[ℋ][x], {x, 0, 1, 0.01},
  PlotRange → All, PlotStyle → Black, PlotLabel → lab]]
```

Out[15]=



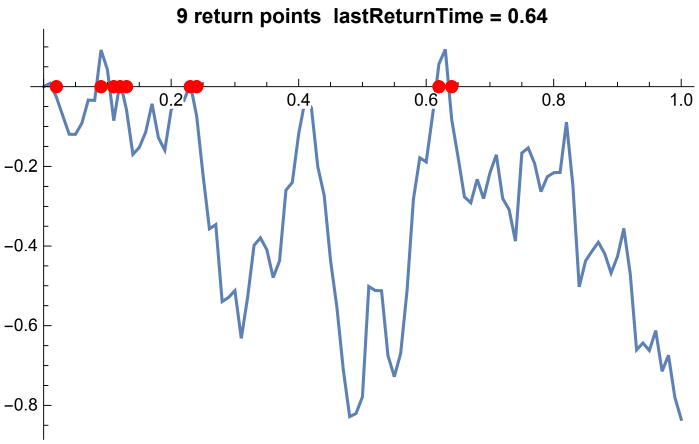
Second arcsine law: arcsine law of last returns

I illustrate the calculation of the return times for the rw.

In[17]:=

```
Module[{nSteps = 100,  data, paths, returnTimes, returnPoints, lab},
 data = PLAsBrownianMotion[nSteps]["Paths"]⟦1⟧;
 returnTimes  = Select[Partition[data, 2, 1], (#⟦1, 2⟧ #⟦2, 2⟧  < 0 ) &]⟦All, 2, 1⟧;
 returnPoints = Point[{#, 0}] & /@ returnTimes;
 lab = Stl@StringForm["`` return points  lastReturnTime = ``",
    Length[returnTimes], If[Length[returnTimes] > 0, N[returnTimes⟦-1⟧], 0] ];
 ListLinePlot[data, Epilog → {Red, PointSize[0.02], returnPoints}, PlotLabel → lab]]
```

Out[17]=



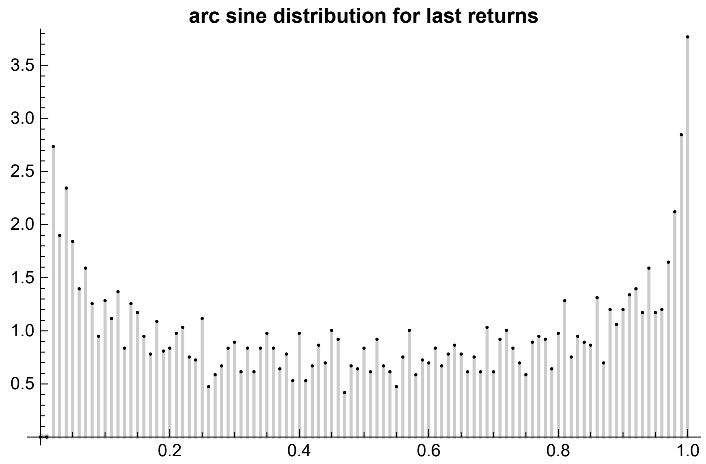
In[18]:=

```
Clear[LastReturnTime];
LastReturnTime[path_] :=
 Module[{returnTimes},
  returnTimes  = Select[Partition[path, 2, 1], (#⟦1, 2⟧ #⟦2, 2⟧  < 0 ) &]⟦All, 2, 1⟧;
  If[Length[returnTimes] > 0, N[returnTimes⟦-1⟧], 0]]
```

In[20]:=

```
Module[{nSteps = 100 , nCases = 4000, nBins = 100, data, ℋ, lab},
 data = Table[LastReturnTime[PLAsBrownianMotion[nSteps]["Paths"]⟦1⟧], {nCases}];
 data = Select[data, (# > 0) &];
 ℋ = HistogramDistribution[data, nBins];
 lab = Stl@StringForm["arc sine distribution for last returns"];
 DiscretePlot[PDF[ℋ][x], {x, 0, 1, 0.01},
  PlotRange → All, PlotStyle → Black, PlotLabel → lab] ]
```

Out[20]=



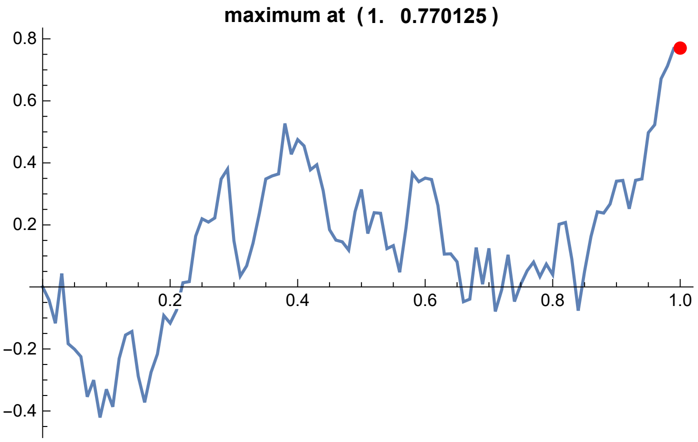
Third arcsine law: arcsine law of maxima

I illustrate the calculation of the return times for the rw.

In[24]:=

```
Module[{nSteps = 100,  data, paths, returnTimes,
  returnPoints, posMax, maxCoordinates, maxPoint, lab},
 data = PLAsBrownianMotion[nSteps]["Paths"]⟦1⟧;
 maxCoordinates = MaximalBy[data, Last];
 lab = Stl@StringForm["maximum at  ``", N@maxCoordinates ];
 ListLinePlot[data, PlotLabel → lab,
  Epilog → {Red, PointSize[0.02], Point[maxCoordinates]}]]
```

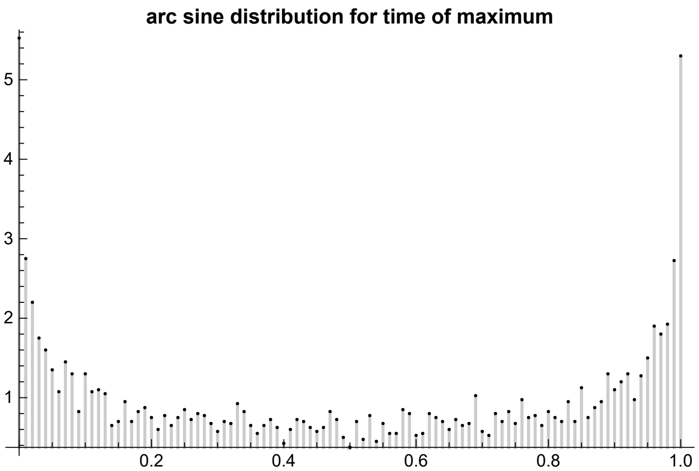
Out[24]=



In[25]:=

```
Module[{nSteps = 100 , nCases = 4000, nBins = 100, data, ℋ, lab},
 data =
  Table[MaximalBy[PLAsBrownianMotion[nSteps]["Paths"][[1]], Last][[1, 1]], {nCases}];
 ℋ = HistogramDistribution[data, nBins];
 lab = Stl@StringForm["arc sine distribution for time of maximum"];
 DiscretePlot[PDF[ℋ][x], {x, 0, 1, 0.01},
  PlotRange → All, PlotStyle → Black, PlotLabel → lab]]
```

Out[25]=



Bournoulli trial paths

I reconsider the arcsine distributions for Bernoulli trial paths. These model trades for which there is a relatively high probability p of winning 1 unit and a corresponding relative low probability q of losing a larger amount. I consider p = 4/5 to win 1 unit and q = 1/5 to lose 4 units. The expectation is zero.

First arcsine law: arcsine law of leads

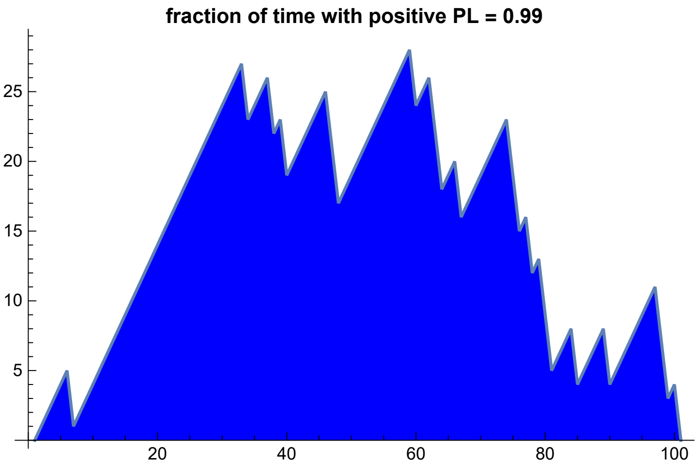
In[5]:=

```
Clear[FractionalTimePositivePL];
FractionalTimePositivePL[n_] :=
 Count[PLAsBrownianMotion[n]["Values"], _?Positive] / n // N
```

In[26]:=

```
Module[{q = 1/5, nTrades = 100, W0 = 0, p, data, fractionTimePositive, lab},
 p = 1 - q;
 data = BournoulliPath[p, nTrades, W0];
 fractionTimePositive = N[Count[data, _?Positive] / nTrades];
 lab =
  Stl@StringForm["fraction of time with positive PL = ``", fractionTimePositive];
 ListLinePlot[data, Filling → Axis, FillingStyle → {White, Blue}, PlotLabel → lab]]
```

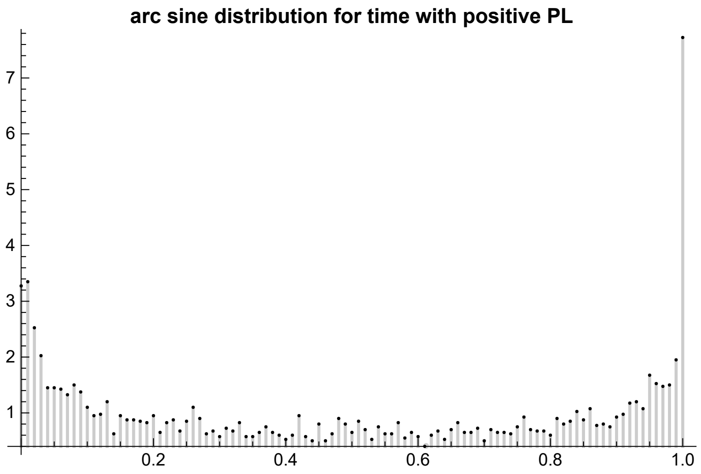
Out[26]=



In[27]:=

```
Module[{q = 1/5 , W0 = 0, nTrades = 100,
  nCases = 4000, nBins = 100, p, path , data, ℋ, lab},
 p = 1 - q;
 data = Table[data = BournoulliPath[p, nTrades, W0];
   N[Count[data, _?Positive] / nTrades], {nCases}];
 ℋ = HistogramDistribution[data, nBins];
 lab = Stl@StringForm["arc sine distribution for time with positive PL"];
 DiscretePlot[PDF[ℋ][x], {x, 0, 1.0, 0.01},
  PlotRange → All, PlotStyle → Black, PlotLabel → lab]]
```

Out[27]=

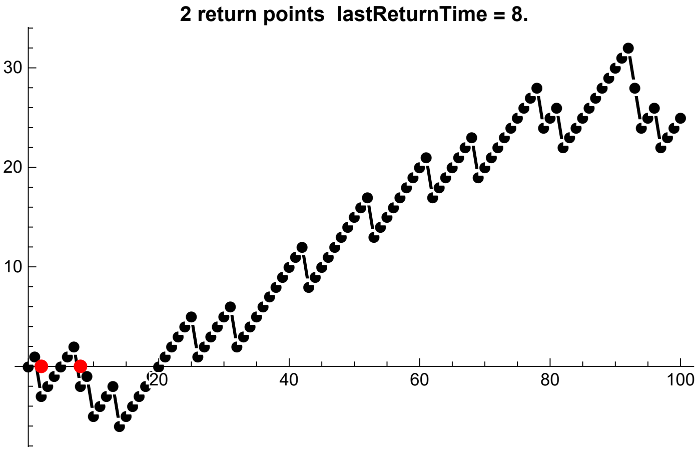


Second arcsine law: arcsine law of last returns

In[28]:=

```
Module[{q = 1/5, nTrades = 100,
  W0 = 0, p, data, paths, returnTimes, returnPoints, lab},
 p = 1 - q;
 data = BournoulliPath[p,  nTrades, W0];
 data = Transpose[{Range[0, nTrades], data}];
 returnTimes  = Select[Partition[data, 2, 1], (#⟦1, 2⟧ #⟦2, 2⟧  < 0 ) &]⟦All, 2, 1⟧;
 returnPoints = Point[{#, 0}] & /@ returnTimes;
 lab = Stl@StringForm["`` return points  lastReturnTime = ``",
    Length[returnTimes], If[Length[returnTimes] > 0, N[returnTimes⟦-1⟧], 0] ];
 ListLinePlot[data, Epilog → {Red, PointSize[0.02], returnPoints},
  PlotLabel → lab, PlotMarkers → Automatic, PlotStyle → Black]]
```

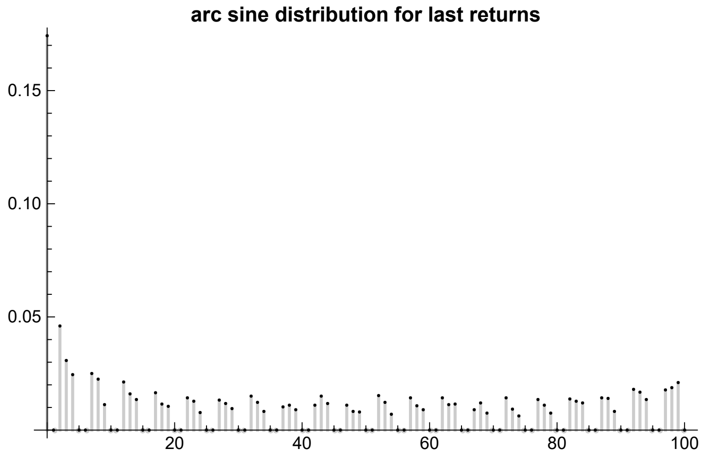
Out[28]=



In[29]:=

```
Module[{q = 1/5 , W0 = 0, nTrades = 100,
  nCases = 4000, nBins = 100, p, path , data, ℋ, lab},
 p = 1 - q;
 data = Table[path = BournoulliPath[p,  nTrades, W0];
   path = Transpose[{Range[0, nTrades], path}];
   LastReturnTime[path], {nCases}];
 ℋ = HistogramDistribution[data, nBins];
 lab = Stl@StringForm["arc sine distribution for last returns"];
 DiscretePlot[PDF[ℋ][x], {x, 0, nTrades},
  PlotRange → All, PlotStyle → Black, PlotLabel → lab] ]
```

Out[29]=



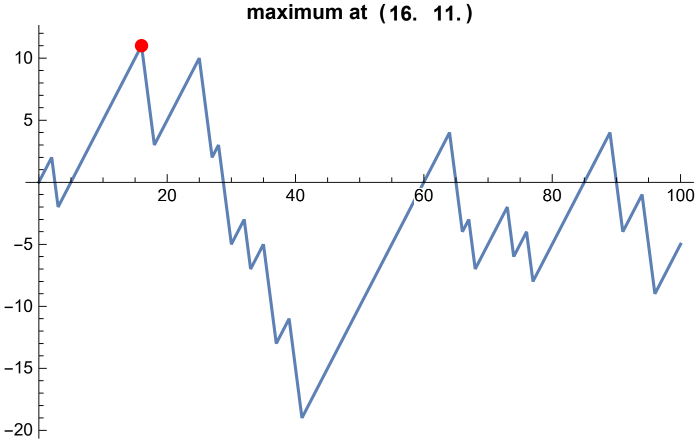
Third arcsine law: arcsine law of maxima

I illustrate the calculation of the maximum PL times for the rw.

In[30]:=

```
Module[{q = 1/5, nTrades = 100, W0 = 0, p,  data, paths,
  returnTimes, returnPoints, posMax, maxCoordinates, maxPoint, lab},
 p = 1 - q;
 data = BournoulliPath[p,  nTrades, W0];
 data = Transpose[{Range[0, nTrades], data}];
 maxCoordinates = MaximalBy[data, Last];
 lab = Stl@StringForm["maximum at  ``", N@maxCoordinates ];
 ListLinePlot[data, PlotLabel → lab,
  Epilog → {Red, PointSize[0.02], Point[maxCoordinates]}]]
```

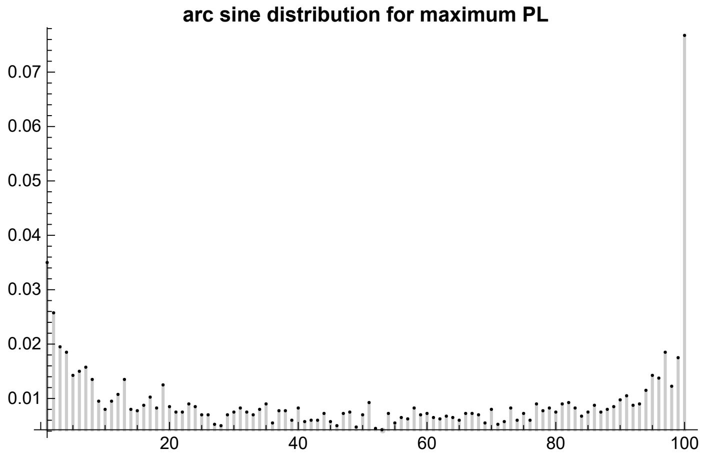
Out[30]=



In[31]:=

```
Module[{q = 1/5 , W0 = 0, nTrades = 100,
  nCases = 4000, nBins = 100, p, maxCoordinates, data, ℋ, lab},
 p = 1 - q;
 data = Table[data = BournoulliPath[p, nTrades, W0];
   data = Transpose[{Range[0, nTrades], data}];
   maxCoordinates = MaximalBy[data, Last];
   maxCoordinates ⟦1, 1⟧ , {nCases}];
 ℋ = HistogramDistribution[data, nBins];
 lab = Stl@StringForm["arc sine distribution for maximum PL"];
 DiscretePlot[PDF[ℋ][x], {x, 1, 100},
  PlotRange → All, PlotStyle → Black, PlotLabel → lab]]
```

Out[31]=



Functions

In[34]:=

```
Clear[LastReturnTime];
LastReturnTime[path_] :=
 Module[{returnTimes},
  returnTimes = Select[Partition[path, 2, 1], (#⟦1, 2⟧ #⟦2, 2⟧ < 0 ) &]⟦All, 2, 1⟧;
  If[Length[returnTimes] > 0, N[returnTimes⟦-1⟧], 0]]
```

In[7]:=

```
Clear[BournoulliPath];
BournoulliPath[p_, n_, W0_] :=
 Module[{q = 1/5, variates, pathB, loss},
  loss = 1 - 1/q;
  variates = RandomVariate[BernoulliDistribution[p], n] /. 0 → loss;
  pathB = FoldList[Plus, 0, variates]]
```