Statistics 581, Chapter 2

Empirical Distribution Function and Empirical Process Figures

Wellner; 10/24/2008

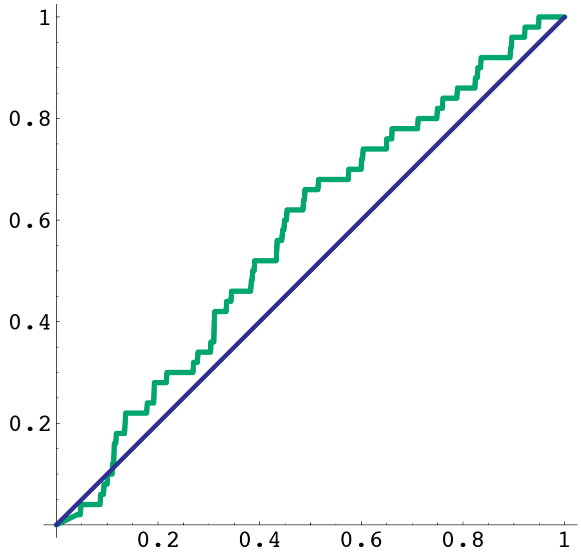

Figure 1: Uniform Empirical Distribution Function, $n = 50$.

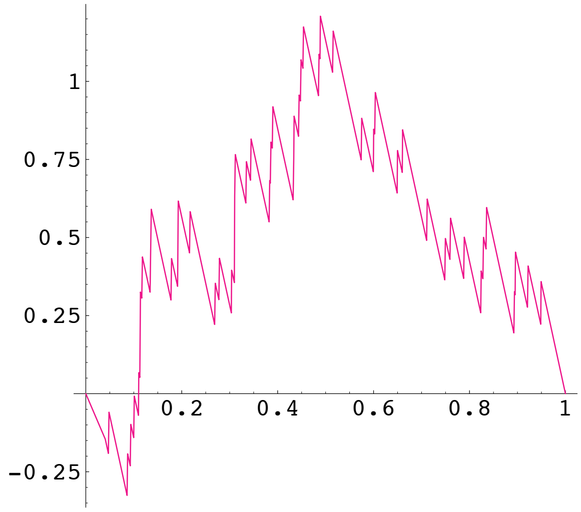

Figure 2: Uniform Empirical Process, $n = 50$.

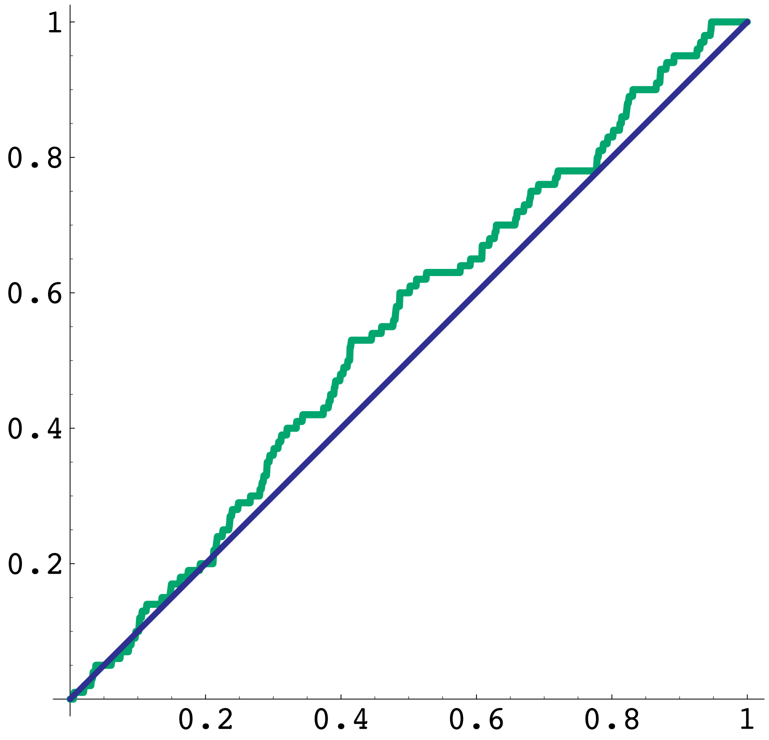

Figure 3: Uniform Empirical Distribution Function, $n = 100$.

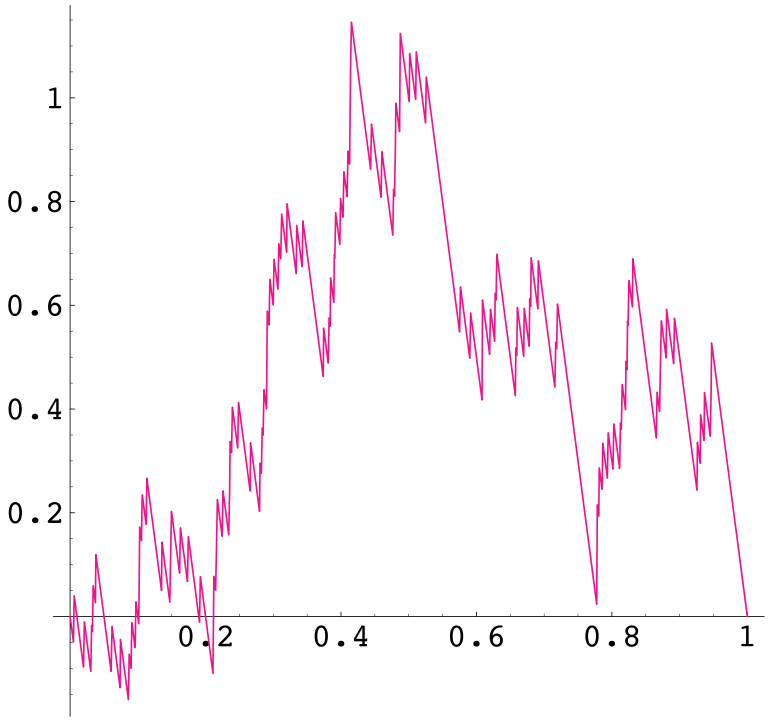

Figure 4: Uniform Empirical Process, $n = 100$.

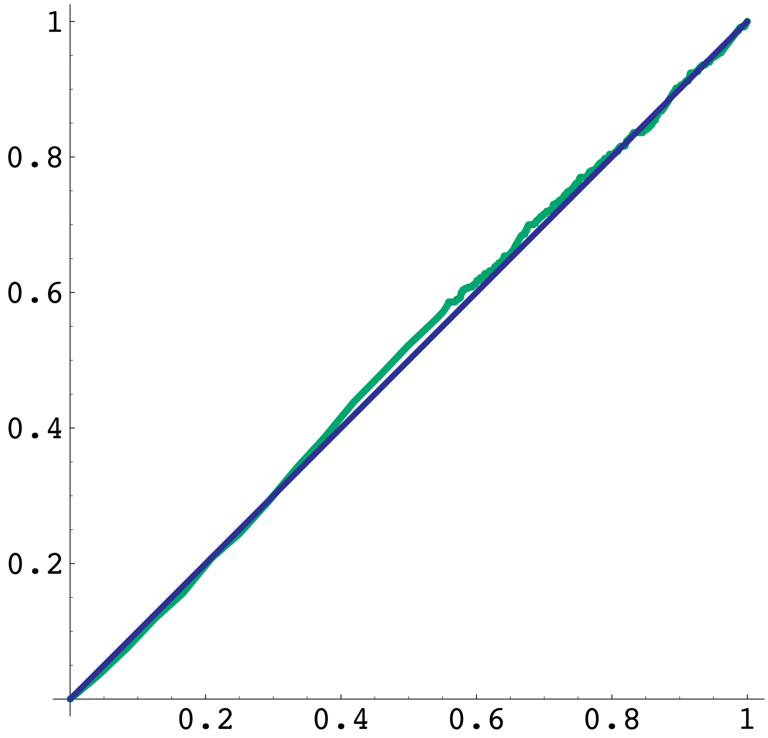

Figure 5: Uniform Empirical Distribution Function, $n = 500$.

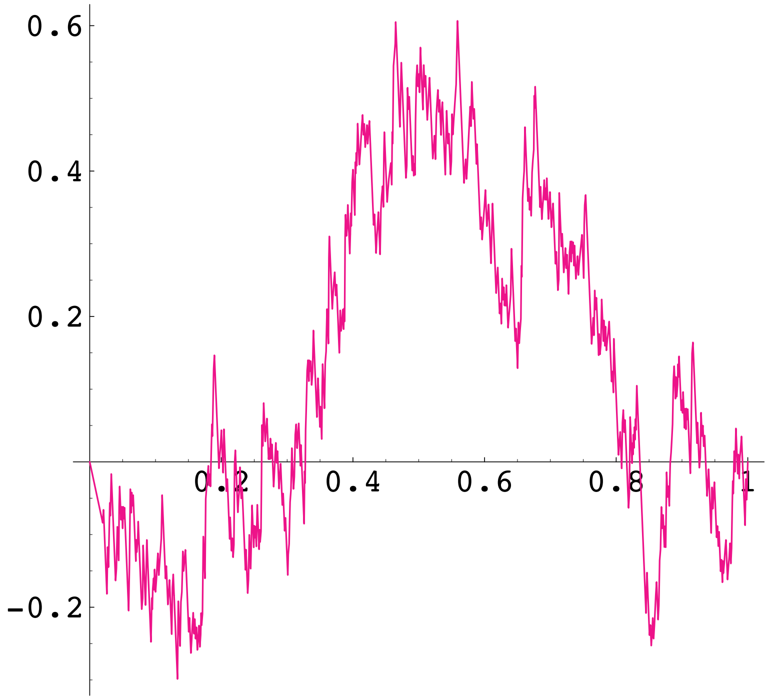

Figure 6: Uniform Empirical Process, $n = 500$.

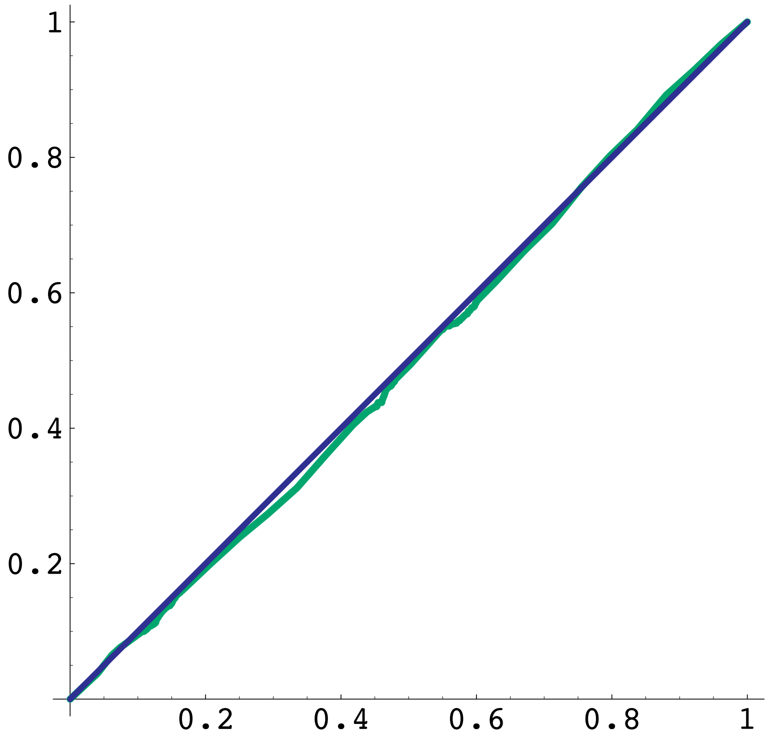

Figure 7: Uniform Empirical Distribution Function, $n = 1000$.

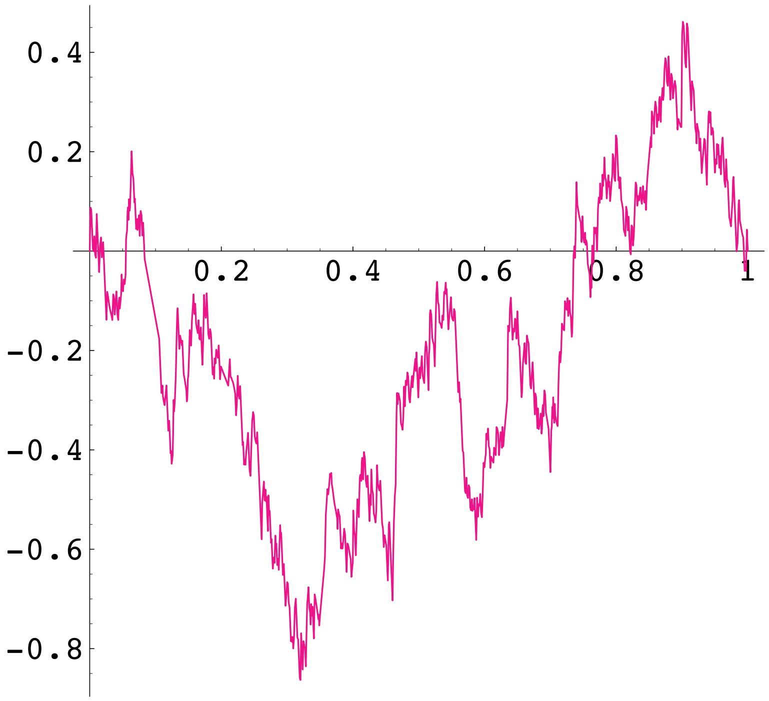

Figure 8: Uniform Empirical Process, $n = 1000$.

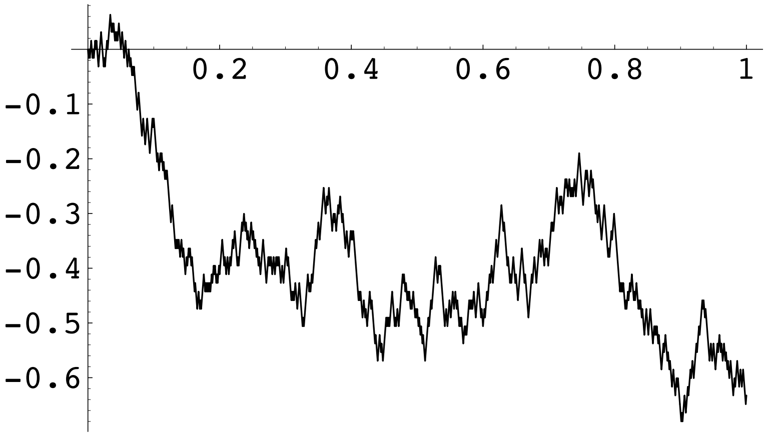

Figure 9: Partial Sum Process, Bernoulli(1/2) $n = 1000$.

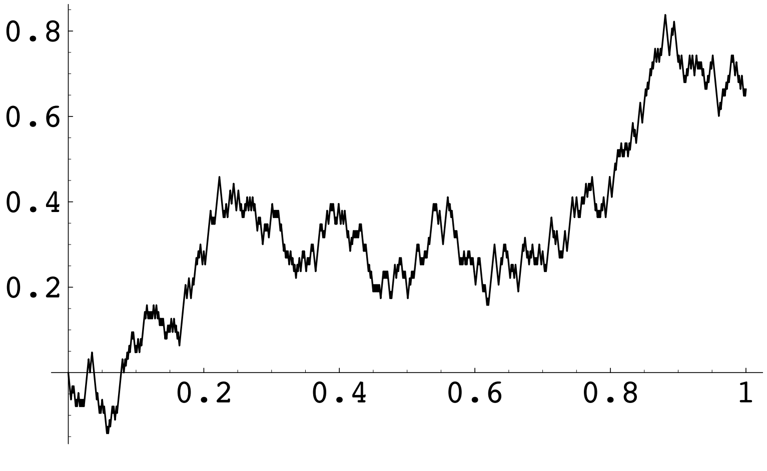

Figure 10: Partial Sum Process, Bernoulli(1/2) $n = 1000$.

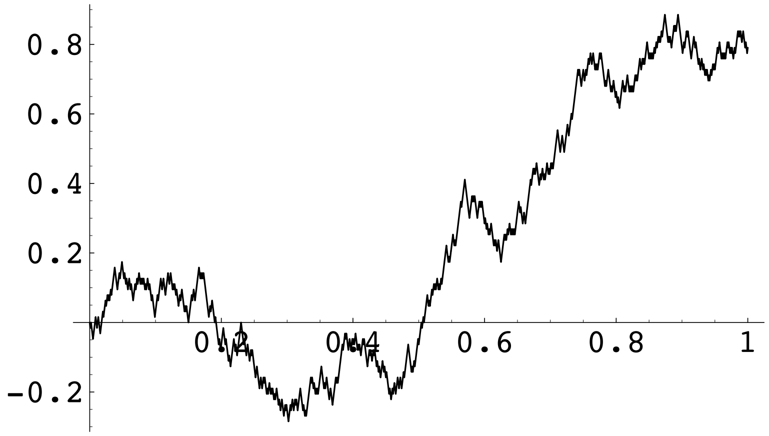

Figure 11: Partial Sum Process, Bernoulli(1/2) $n = 1000$.

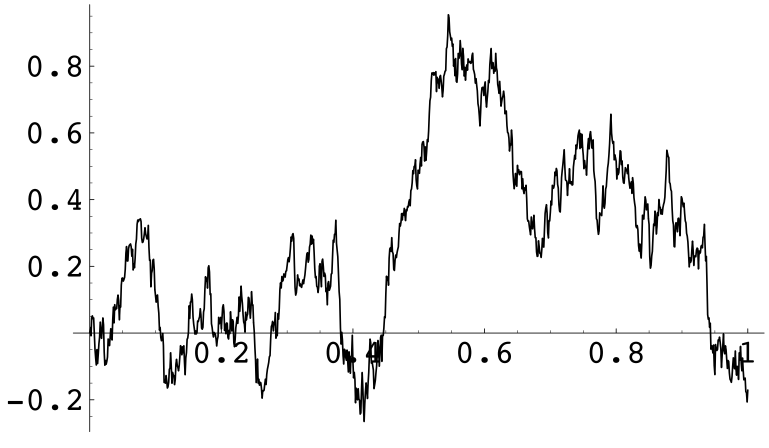

Figure 12: Partial Sum Process, $N(0, 1)$, $n = 1000$.

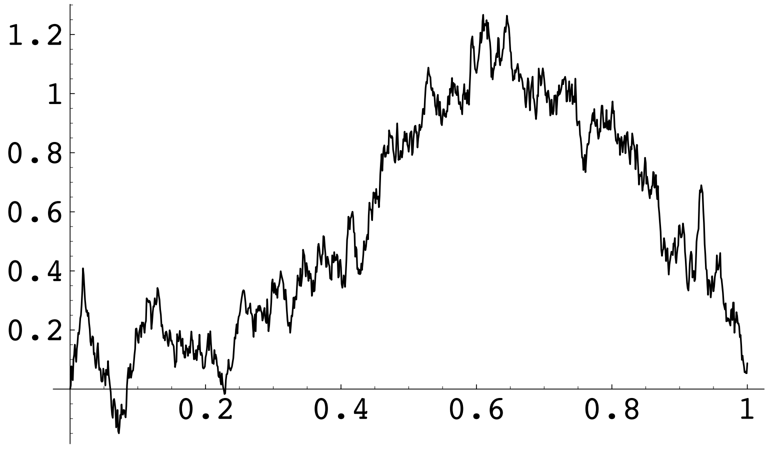

Figure 13: Partial Sum Process, $N(0,1)$, $n = 1000$.

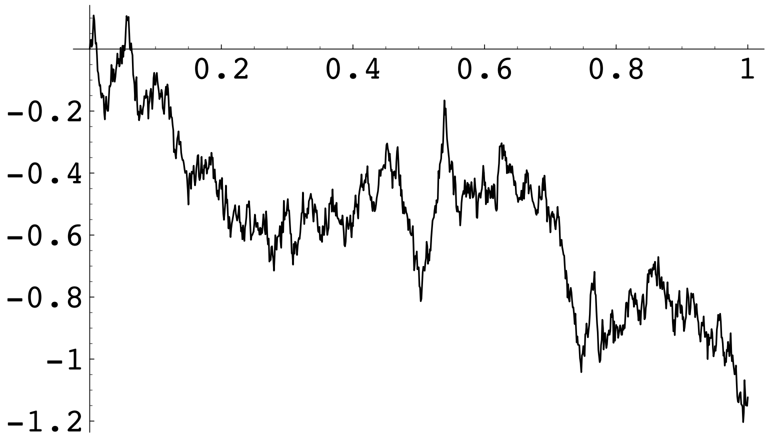

Figure 14: Partial Sum Process, $N(0,1)$, $n = 1000$.

Mathematica Code for Figures 1-8:

```
n = 50
x = Table[Random[ ], {n}]
yy = Sort[x]
zz = Table[k , {k,1,n}]*(1/n)  //N
Table[{yy[[i]],zz[[i]]}, {i,1,n}]
Delta[x_] := 0 /;  x < 0
Delta[x_] := 1 /;  x >= 0
DE[x_] := Table[{Delta[yy[[i]] - x]}, {i,1,n}]
EDF[n_, x_] := 1. - Apply[Plus,DE[x]]*(1/n)  //N
Funif[x_] := x
EMP[n_, x_] := Sqrt[n]*(EDF[n,x] - Funif[x])
Plot[{EDF[n,x], Funif[x]}, {x,0,1},
    PlotStyle -> {
     {Thickness[1/100],RGBColor[0.000, 1.000, 0.196],Dashing[{}]},
     {Thickness[1/120],RGBColor[0,0,1],Dashing[{}]}
    }, AspectRatio->1.0
    ]
Plot[EMP[n,x], {x,0,1},
PlotStyle ->
RGBColor[1.000, 0.032, 0.948],
AspectRatio->1.0
]
```

Mathematica Code for Figures 9-14:

```
<< Statistics‘ContinuousDistributions‘
<< Statistics‘DiscreteDistributions‘
<< Statistics‘DescriptiveStatistics‘
gdist := NormalDistribution[0, 1]
bdist := BernoulliDistribution[.5]
PartialSums[n_Integer] := {
    xg1 = Table[Random[gdist], {n}];
    sg1 = FoldList[Plus, 0, xg1]/Sqrt[n];
    xg2 = Table[Random[bdist] - .5, {n}];
    sg2 = FoldList[Plus, 0, xg2]/Sqrt[n];
    zz = Table[k, {k, 0, n}]*(1/n) // N;
    dd = Table[1, {k, 0, n}]*(1/n)}
PartialSums[1000];
xg1;
xg2;
sg1;
sg2;
J1 = Transpose[{zz, sg1}];
J2 = Transpose[{zz, sg2}];
ListPlot[J1, PlotJoined -> True]
ListPlot[J2, PlotJoined -> True]
```