Functions

In[1]:=
```
<< HypothesisTesting`
```

In[2]:=
```
TwoBenfordsLawProb[d_] := Module[{},
  N[Sum[Log[10, 1 + (10 k + d)^-1], {k, 1, 9}]]
 ]
twoBLProbs = Map[TwoBenfordsLawProb, Range[0, 9]]
```

Out[3]=
```
{0.119679, 0.11389, 0.108821, 0.10433, 0.100308,
 0.0966772, 0.0933747, 0.090352, 0.0875701, 0.0849974}
```

In[4]:=
```
GetSignificantDigit[num_, dig_] := Module[{digits},
   digits = RealDigits[num];
   If[digits[[2]] ≠ 1,
    Return[digits[[1, dig]]],
    Return[digits[[1, 1]]]]
  ];
```

In[5]:=
```
DigitProbs[data_, digitPos_] := Module[{},
  Transpose[MapAt[N[# / Length[data]] &,
    Transpose[Sort[Tally[Map[GetSignificantDigit[#, digitPos] &, data]]]]
    ,
    2]]]
```

In[6]:=
```
DigitTally[data_, digitPos_] := Module[{},
  Sort[Tally[Map[GetSignificantDigit[#, digitPos] &, data]]]]
```

In[7]:=
```
X2B2[list_] := Module[{tallySecondDigit, totalSecondDigit, x2B2},
  tallySecondDigit = DigitTally[list, 2];
  totalSecondDigit = Total[tallySecondDigit[[All, 2]]];
  x2B2 = Sum[(tallySecondDigit[[i, 2]] - totalSecondDigit * twoBLProbs[[i]])^2 /
      (totalSecondDigit * twoBLProbs[[i]]), {i, 1, 10}]
 ]
```

In[8]:=
```
St[alpha_, T_] := Module[{},
  Map[# * alpha / T &, Range[T]]
 ]
```

In[9]:=
```
PValue[stat_] := Module[{},
  Last[ChiSquarePValue[stat, 9, TwoSided → False]]]
```

In[10]:=
```
PValueMap[stats_] := Module[{},
  Map[PValue[#] &, stats]
 ]
```

In[11]:=
```
Stats[data_] := Module[{},
  {"Mean:" <> ToString[N[Mean[data]]],
   "Median:" <> ToString[N[Median[data]]], "Min:" <> ToString[Min[data]],
   "Max:" <> ToString[Max[data]], "Skewness:" <> ToString[N[Skewness[data]]]}
 ]
```

In[12]:=
```
BLHistogram[data_] :=
 Module[{}, BarChart[Transpose[{twoBLProbs, DigitProbs[data, 2][[All, 2]]}],
   ChartLegends → {"2BL", "Data"}, ChartStyle → {Green, Blue},
   ChartLabels → {Range[0, 9], None},
   PlotLabel → "Second Digit Benford Expected Frequency", ImageSize → Medium]]
```

In[13]:=
```
DistrictVoteGenerator[cand_, weights_, numVotes_] :=
 Module[{votes, missing, tally, i},
  votes = RandomChoice[weights → cand , numVotes];
  tally = Tally[votes];
  missing = Flatten[Position[Map[MemberQ[votes, #] &, cand], False]];
  For[i = 1, i ≤ Length[missing], i++,
   tally = Append[tally, {cand[[missing[[i]]]], 0}];];
  Flatten[Map[Cases[tally, {#, _}][[All, 2]] &, cand]]
 ]
```

In[14]:=
```
DataLogPlot[data_] := Module[{log10},
  log10 = Map[N[Log[10, #]] &, data];
  Histogram[log10, Automatic, "Probability"]
 ]
```

In[15]:=
```
MechA[size_, mf_, lgp_, hgp_, lb_, ha_] := Module[{lgb, hgb, mgb, p3, q, pf, votes},
 lgb = Exp[lgp] / (Exp[lgp] + Exp[hgp] + 1);
  hgb = Exp[hgp] / (Exp[lgp] + Exp[hgp] + 1);
  mgb = 1 / (Exp[lgp] + Exp[hgp] + 1);
 p3 = {RandomVariate[BetaDistribution[1 / 2, lb]],
    mf, RandomVariate[BetaDistribution[ha, 1 / 2]]};
 q = RandomVariate[UniformDistribution[{0, 1}]];
 pf = {q * lgb, mgb, (1 - q) * hgb};
  votes = Round[Total[size * p3 * pf / Total[pf]]];
 {votes, size - votes}]
```

In[16]:=
```
AdjustedPValue[p_] := Module[{adjusted, T, pSorted, j},
  pSorted = Sort[p];
  T = Length[p];

  adjusted = {};

  For[j = 1, j ≤ T, j++,
   adjusted = Append[adjusted, Min[Map[Min[T/# * pSorted[[#]], 1] &, Range[j, T]]]]
  ];

  adjusted
 ]
```

Random Counts

Texas Districts

In[17]:=
```
nd = NotebookDirectory[];
in = Import[nd <> "Texas//Texas_Normal_Vote_2002-2010.txt", "Data"];
labels = in[[1]];
data = in[[2 ;;]];
data // Length
posReg = First[First[Position[labels, "t_Registered_Voters_2010"]]];
posTurnOut = First[First[Position[labels, "t_Turnout_2010"]]];
```

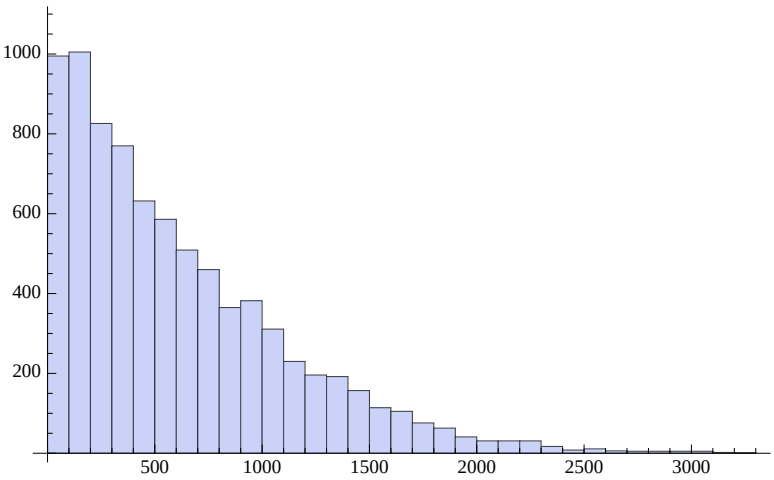
Out[21]= 8400

In[24]:=
```
hot = data[[All, posTurnOut]];
hot2 = DeleteCases[hot, 0];
hot2 // Length
Stats[hot2]
Histogram[hot2]
```

Out[26]= 8173

Out[27]= {Mean:610.249, Median:476., Min:1, Max:3283, Skewness:1.25071}

Out[28]=



- **Random Choice with Weights**
- **Mechanism A**

Understanding Equations

Policy Based Vote

- **Functions**

In[29]:=
```
UniRand[] := Module[{},
  RandomVariate[UniformDistribution[{0, 1}]]
 ]
```

In[30]:=
```
PoliticalPlayer[numPolices_] := Module[{},
  Table[UniRand[], {x, numPolices}]
 ]
```

In[31]:=
```
P[b_, x_, k_] := Module[{},
  Sum[b[[j]] * (x[[j]] - k[[j]])^2, {j, 1, Length[k]}]
 ];
```

In[42]:=
```
VoterChoice[voterI_, b_, candidates_] := Module[{scores},
  scores = Map[P[b, #, voterI] &, candidates];
  First[Ordering[scores, 1]]
 ]
```

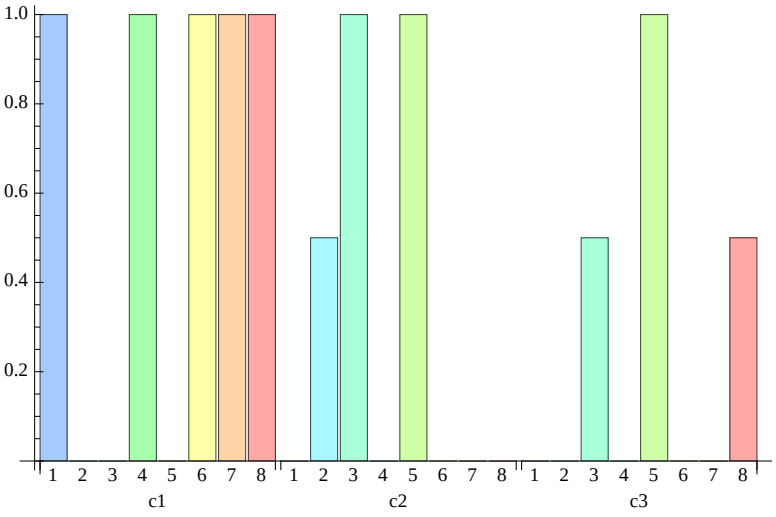
In[33]:=
```
DistrictTally[voters_, b_, candidates_] := Module[{},
  Sort[Tally[Map[VoterChoice[#, b, candidates] &, voters]]]
 ]
```

In[34]:=
```
RandomVoters[nVoters_, candidates_, nPolices_, b_] :=
 Module[{voter, nCandidates, tally, range, freeq, posq},
  voter = Table[PoliticalPlayer[nPolices], {x, nVoters}];
  nCandidates = Length[candidates];
  tally = DistrictTally[voter, b, candidates];
  If[Length[tally] == nCandidates, tally[[All, 2]],
   range = Range[nCandidates];
   freeq = Map[FreeQ[tally[[All, 1]], #] &, range];
   posq = Flatten[Position[freeq, True]];
   Sort[Join[tally, Map[{#, 0} &, range[[posq]]]]][[All, 2]]
  ]
 ]

ElectionResults[precictVotes_] := Module[{totalsPerCand, totalVotes},
  totalsPerCand = Map[Total, Transpose[distSub]];
  totalVotes = Total[totalsPerCand];
  N[totalsPerCand / totalVotes]
 ]
```

■ Random Candidates and Voters

```
nCandidates = 3;
cand = Range[nCandidates];
nPolices = 8;
b = Table[1, {x, nPolices}];
(*candidate=Table[PoliticalPlayer[nPolices],{x,nCandidates}];*)
candidate = {{1, 0, 0, 1, 0, 1, 1, 1}, {0, .5, 1, 0, 1, 0, 0, 0}, {0, 0, .5, 0, 1, 0, 0, .5}};
(*
c1=Left wing;
c2=Right wing;
c3=Others;

1: Should abortion remain a legal option in America?;
2: Should law enforcement be allowed to use racial profiling?;
3: Should the federal deficit be reduced without raising any taxes?;
4: Are the March 2010 federal health
  care reform laws ("Obamacare") good for America?;
5: Should state and local law enforcement be empowered
  to enforce federal immigration laws?;
6: Should gay marriage be legal?;
7: Should marijuana be a medical option?;
8: Should the wealthiest 1% of Americans be taxed more heavily?;
Source→http://2012election.procon.org/view.source-summary-chart.php;
*)
BarChart[candidate, ChartLabels → {{"c1", "c2", "c3"}, Range[nPolices]}]
```

Out[291]=



In[319]:=
```
AbsoluteTiming[
   dist = Map[RandomVoters[#, candidate, nPolices, b] &, hot2];
  ][[1]] / 60
```

Out[292]= 10.56840897

```
distSub > (nd <> "dist1.txt")
```

- **Filtering Small Precincts**
- **Not Filtering**

In[295]:=
```
bool = Map[Boole[# > 0] &, hot2];
distSub = Pick[dist, bool, 1];
```

In[314]:=
```
chiStat = Map[X2B2, Transpose[distSub]];
pValue = Map[PValue, chiStat];
adjustedPValue = AdjustedPValue[pValue];
electionResults = ElectionResults[distSub];
TableForm[Transpose[{cand, chiStat, pValue, adjustedPValue, electionResults}]]
```

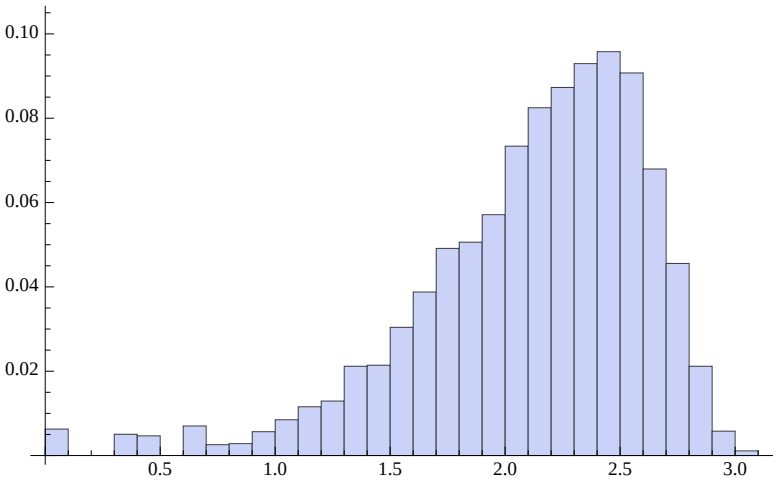
Out[318]//TableForm=

1	6.63423	0.324859	0.324859	0.339685
2	6.35244	0.295808	0.324859	0.255462
3	5.1855	0.182153	0.324859	0.404853

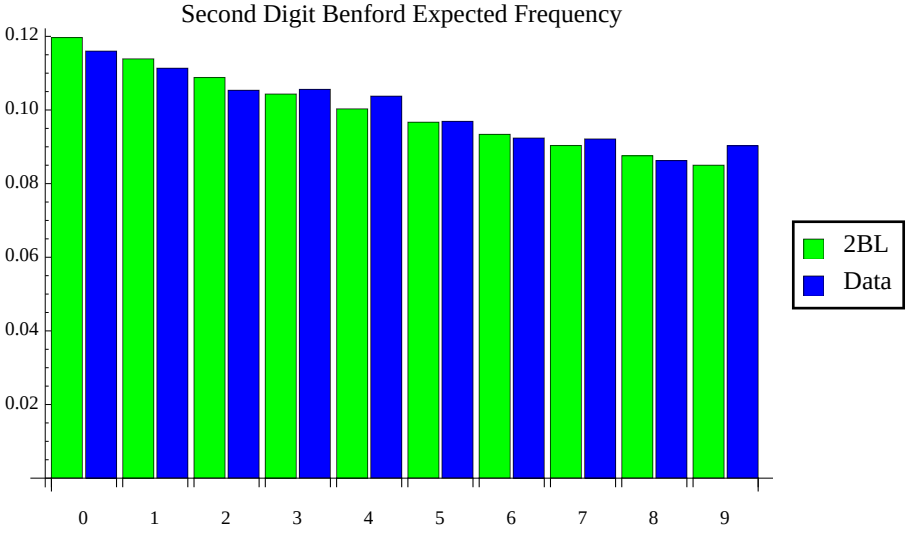
In[305]:=
```
focus = dist[[All, 1]];
Stats[focus]
DataLogPlot[focus]
BLHistogram[focus]
```

Out[306]= {Mean:207.293, Median:162., Min:0, Max:1125, Skewness:1.24311}

Out[307]=



Out[308]=



Test

```
Plot[PDF[ExtremeValueDistribution[EulerGamma, Pi^2/6], x], {x, -8, 12}]
```