Michel Fodje's epr-simple simulation translated from Python to Mathematica by John Reed 13 Nov 2013
Using Joy Christian's complete state parameters

Set run time parameters, initialize arrays

In[5]:=
```
trials = 5 000 000;
trialDeg = 360;
```

In[7]:=
```
aliceDeg = ConstantArray[0, trials];
bobDeg = ConstantArray[0, trials];
aliceDet = ConstantArray[0, trials];
bobDet = ConstantArray[0, trials];
```

In[11]:=
```
nPP = ConstantArray[0, trialDeg];
nNN = ConstantArray[0, trialDeg];
nPN = ConstantArray[0, trialDeg];
nNP = ConstantArray[0, trialDeg];
nAP = ConstantArray[0, trialDeg];
nBP = ConstantArray[0, trialDeg];
nAN = ConstantArray[0, trialDeg];
nBN = ConstantArray[0, trialDeg];
```

Complete State Selection

In[19]:=
```
n = 1; (*n=1 for spin 1/2; n=2 for spin 1*)
test1[angle_, e_, λ_] := Module[{c, out},
  If[Abs[angle - e] < π, d = 1, d = -1];  (*Polarizer for A Station*)
  c = -(-1)^n Cos[n * angle * d];
  If[λ ≥ Abs[c], out = 0, out = Sign[c]];
  out]
test2[angle_, e_, λ_] := Module[{c, out},
  If[Abs[angle - e] < π, d = 1, d = -1];  (*Polarizer for B Station*)
  c = (-1)^n Cos[n (angle * d)];
  If[λ ≥ Abs[c], out = 0, out = Sign[c]];
  out]
```

Generate Particle Data

In[22]:=
```
Do[
 t = RandomReal[{0, π}];
 λ = (2 / Sqrt[1 + 3 t / π] - 1) * 1;
 eLeft = RandomReal[{0, 2 π}]; (*select random e or fixed e*)
 (*eLeft=0π/8;*)
 eRight = eLeft + n * π;
 aliceAngle = RandomReal[{0, 2 π}];
 aliceDeg[[i]] = aliceAngle / Degree;
 bobAngle = RandomReal[{0, 2 π}];
 bobDeg[[i]] = bobAngle / Degree;
 aliceDet[[i]] = test1[aliceAngle, eLeft, λ];
 bobDet[[i]] = test2[bobAngle, eRight, λ],
 {i, trials}]
```

Statistical Analysis of Particle Data

In[23]:=
```
Do[
 θ = Round[aliceDeg[[i]] - bobDeg[[i]]];
 aliceD = aliceDet[[i]]; bobD = bobDet[[i]];
 If[aliceD == 1, nAP[[θ]]++];
 If[bobD == 1, nBP[[θ]]++];
 If[aliceD == -1, nAN[[θ]]++];
 If[bobD == -1, nBN[[θ]]++];
 If[aliceD == 1 && bobD == 1, nPP[[θ]]++];
 If[aliceD == 1 && bobD == -1, nPN[[θ]]++];
 If[aliceD == -1 && bobD == 1, nNP[[θ]]++];
 If[aliceD == -1 && bobD == -1, nNN[[θ]]++],
 {i, trials}]
```

Calculate mean values and plot

In[24]:=
```
pPP = 0; pPN = 0; pNP = 0; pNN = 0;
```

In[25]:=
```
mean = ConstantArray[0, trialDeg];
```

In[26]:=
```
Do[
 sum = nPP[[i]] + nPN[[i]] + nNP[[i]] + nNN[[i]];
If[sum == 0, Goto[jump],
  {pPP = nPP[[i]] / sum;
   pNP = nNP[[i]] / sum;
   pPN = nPN[[i]] / sum;
   pNN = nNN[[i]] / sum;
mean[[i]] = pPP + pNN - pPN - pNP}];
 Label[jump],
 {i, trialDeg}]
```

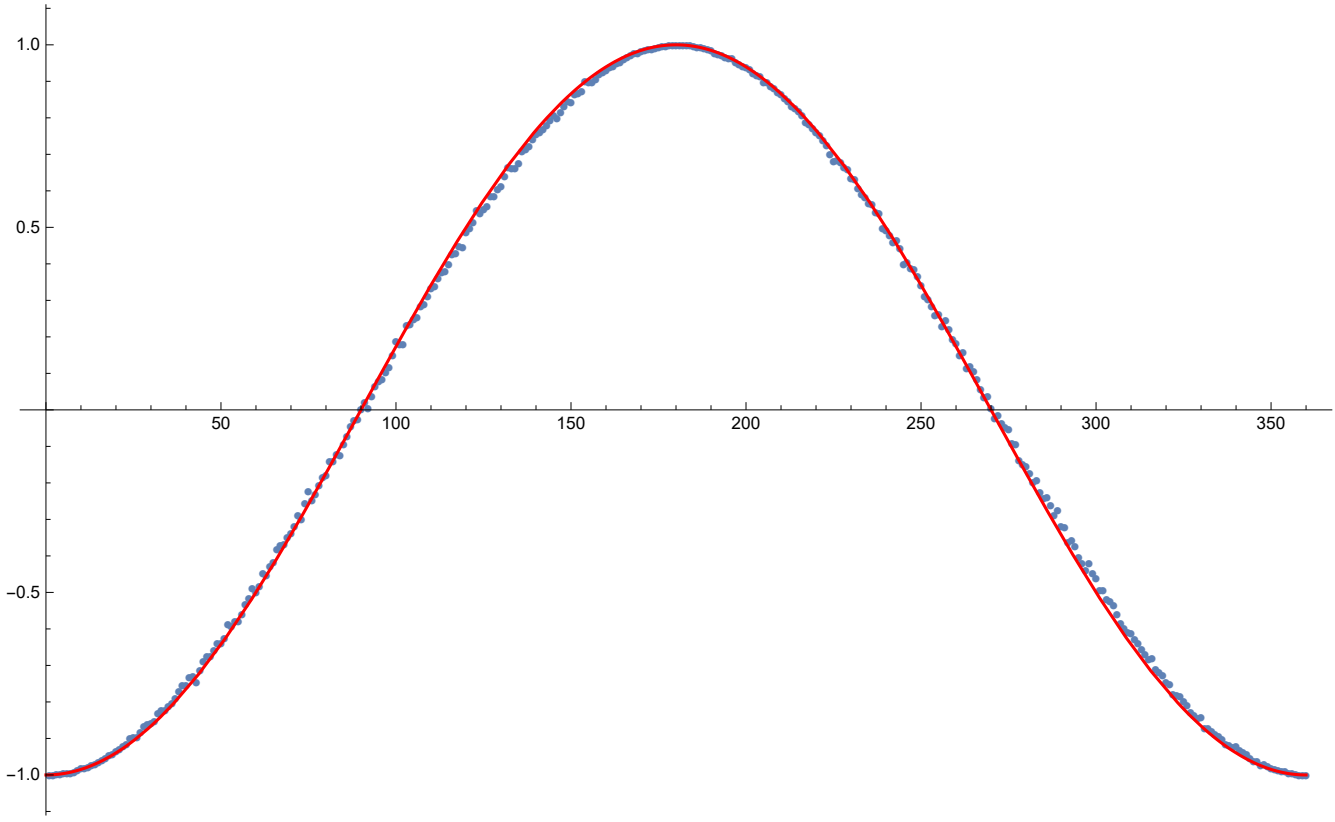
In[27]:=
```
simulation = ListPlot[mean, PlotMarkers → {Automatic, Tiny}];
```

In[28]:=
```
cos = Plot[-Cos[n * x Degree], {x, 0, 360}, PlotStyle → {Red}];
```

Compare mean values with -Cosine Curve and compute averages

In[29]:=
```
Show[simulation, cos]
AveA = N[Sum[aliceDet[[i]], {i, trials}] / trials];
AveB = N[Sum[bobDet[[i]], {i, trials}] / trials];
Print["AveA = ", AveA]
Print["AveB = ", AveB]
PAP = N[Sum[nAP[[i]], {i, trialDeg}]];
PBP = N[Sum[nBP[[i]], {i, trialDeg}]];
PAN = N[Sum[nAN[[i]], {i, trialDeg}]];
PBN = N[Sum[nBN[[i]], {i, trialDeg}]];
PA1 = PAP / (PAP + PAN);
PB1 = PBP / (PBP + PBN);
Print["P(A+)= ", PA1]
Print["P(B+)= ", PB1]
totAB = Sum[nPP[[i]] + nNN[[i]] + nPN[[i]] + nNP[[i]], {i, trialDeg}]
```

Out[29]=




```
AveA = 0.0001422
AveB = -0.0004568
P(A+)= 0.500069
P(B+)= 0.499726
```

Out[42]= 3 148 333