Michel Fodje's epr-simple simulation translated from

Python to Mathematica by John Reed 13 Nov 2013

Modified for QM Local Complete States 6 Aug 2019

No Zeroes in A and B outputs

Set Run Time Parameters, Initialize Arrays and Tables

In[421]:=
```
trialcs = 10000000;
trialDeg = 360;
```

In[423]:=
```
CS1 = Table[{0, 0, 0, 0, 0}, trialcs];
CS2 = Table[{0, 0, 0, 0, 0}, trialcs];
CA = Table[{0, 0}, trialcs];
CB = Table[{0, 0}, trialcs];
s = ConstantArray[0, trialcs];
λ = ConstantArray[0, trialcs];
a1 = ConstantArray[0, trialcs];
b1 = ConstantArray[0, trialcs];
A1 = ConstantArray[0, trialcs];
B1 = ConstantArray[0, trialcs];
```

In[433]:=
```
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];
```

Generate Particle Data with 3 Separate Computers (3 Do Loops)

In[441]:=
```
Do[s[[j]] = Normalize@RandomVariate[NormalDistribution[], 3];
 (*3D uniform random unit vectors*)
 λ[[j]] = RandomChoice[{-1, 1}], {j, trialcs}]
```

In[442]:=
```
Do[a = Normalize@RandomVariate[NormalDistribution[], 3];
 If[a.s[[j]] > 0, s1 = a, s1 = -a];  (*Polarizer Function*)
 A = λ[[j]] (a.(-s1));               (*Measurement Function*)
 CA[[j]] = {a, A}, {j, trialcs}]
```

In[443]:=
```
Do[b = Normalize@RandomVariate[NormalDistribution[], 3];
 If[b.s[[j]] > 0, s2 = b, s2 = -b];
 B = λ[[j]] (b.s2);
 CB[[j]] = {b, B}, {j, trialcs}]
```

Complete States Selection and Statistical Analysis of Particle Data

In[444]:=
```
a1 = CA[[All, 1]]; b1 = CB[[All, 1]]; A1 = CA[[All, 2]]; B1 = CB[[All, 2]];
Do[
 t = RandomReal[{0, π}];
 z = (2/Sqrt[1 + 3 t/π] - 1);
 If[Abs[a1[[j]].s[[j]]] < z || Abs[b1[[j]].s[[j]]] < z, CS = 0, CS = 1];
 CS1[[j]] = {A1[[j]], B1[[j]], a1[[j]], b1[[j]], CS},
 {j, trialcs}]
CS2 = Select[CS1, Last[#] == 1 &];
trials = Length[CS2];
aliceDeg = ConstantArray[0, trials];
bobDeg = ConstantArray[0, trials];
a2 = ConstantArray[0, trials];
b2 = ConstantArray[0, trials];
aliceD1 = ConstantArray[0, trials];
bobD1 = ConstantArray[0, trials];
a2 = CS2[[All, 3]]; b2 = CS2[[All, 4]];
aliceD1 = CS2[[All, 1]]; bobD1 = CS2[[All, 2]];
```

In[456]:=
```
Do[
 x1 = Part[a2[[j]], 1]; y1 = Part[a2[[j]], 2];
 aliceDeg[[j]] = ArcTan[x1, y1];
 x2 = Part[b2[[j]], 1]; y2 = Part[b2[[j]], 2];
 bobDeg[[j]] = ArcTan[y2, x2];
 If[(aliceDeg[[j]] * bobDeg[[j]]) > 0, theta = ArcCos[a2[[j]].b2[[j]]] * 180 / π,
  theta = -ArcCos[a2[[j]].b2[[j]]] * 180 / π + 360];
 θ = Round[theta];
 aliceD = aliceD1[[j]]; bobD = bobD1[[j]];
 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]] ++],
 {j, trials}]
```

Calculate mean values and plot

In[457]:=
```
pPP = 0; pPN = 0; pNP = 0; pNN = 0;
```

In[458]:=
```
mean = ConstantArray[0, trialDeg];
```

In[459]:=
```
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}]
```

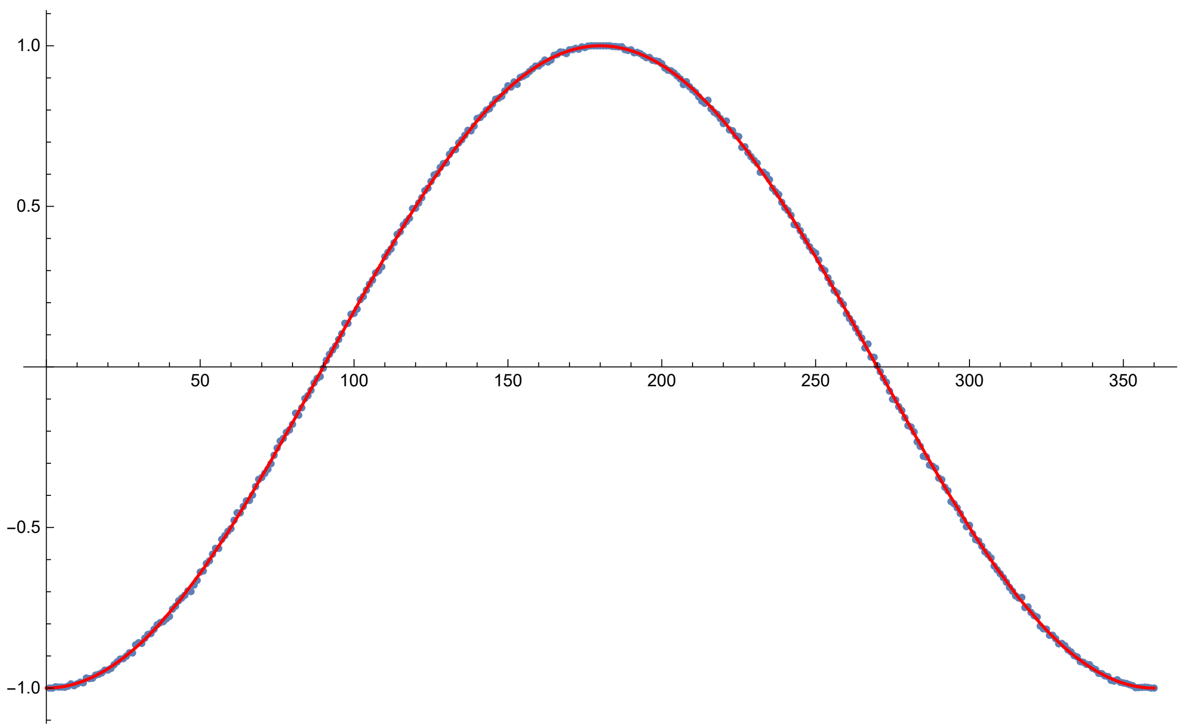
In[460]:=
```
simulation = ListPlot[mean, PlotMarkers → {Automatic, Tiny}];
```

In[461]:=
```
negcos = Plot[-Cos[x Degree], {x, 0, 360}, PlotStyle → {Red}];
```

Compare mean values with -Cosine Curve and compute averages

In[462]:=
```
Show[simulation, negcos]
AveA = N[Sum[A1[[i]], {i, trials}] / trials];
AveB = N[Sum[B1[[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}];
PP = N[Sum[nPP[[i]], {i, trialDeg}] / totAB]
NN = N[Sum[nNN[[i]], {i, trialDeg}] / totAB]
PN = N[Sum[nPN[[i]], {i, trialDeg}] / totAB]
NP = N[Sum[nNP[[i]], {i, trialDeg}] / totAB]
CHSH = Abs[N[mean[[22]]] + N[mean[[67]]] - N[mean[[135]]] + N[mean[[45]]]]
```

Out[462]=



AveA = -4.46557×10^{-6}

AveB = 0.000316279

P(A+) = 0.499855

P(B+) = 0.500018

Out[476]= 0.249789

Out[477]= 0.249916

Out[478]= 0.250066

Out[479]= 0.250229

Out[480]= 2.74495