Michel Fodje's epr-simple simulation translated from Python to Mathematica by John Reed 13 Nov 2013 Modified for QM Local Complete States 26 Jul 2019 2D Vectors

Set run time parameters, initialize arrays

In[828]:=
```
trials = 5 000 000;
trialDeg = 360;
```

In[830]:=
```
aliceDeg = ConstantArray[0, trials];
bobDeg = ConstantArray[0, trials];
aliceDet = ConstantArray[0, trials];
bobDet = ConstantArray[0, trials];
a1 = ConstantArray[0, trials];
b1 = ConstantArray[0, trials];
```

In[836]:=
```
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

In[844]:=
```
Do[
 z = (1/2) Sin[RandomReal[{0, π/2}]]^2;  (*Complete States Parameter*)
 λ = RandomChoice[{-1, 1}];
 s = Flatten[List[Normalize@RandomVariate[NormalDistribution[], 2], 0], 1];
 a = Flatten[List[Normalize@RandomVariate[NormalDistribution[], 2], 0], 1];
 x1 = Part[a, 1]; y1 = Part[a, 2];
 aliceDeg[[i]] = ArcTan[x1, x2];
 a1[[i]] = a;
 b = Flatten[List[Normalize@RandomVariate[NormalDistribution[], 2], 0], 1];
 x2 = Part[b, 1]; y2 = Part[b, 2];
 bobDeg[[i]] = ArcTan[x2, y2];
 b1[[i]] = b;
 If[a.s > 0, s1 = a, s1 = -a];  (*Polarizer Functions*)
 If[b.s > 0, s2 = b, s2 = -b];
 (*Measurement Functions*)
 If[Abs[a.s] < z, A = {0 + 0. i},
  A = λ/2 ((1 0).(PauliMatrix[1] * a[[1]] + PauliMatrix[2] * a[[2]] + PauliMatrix[3] * a[[3]]).
      (PauliMatrix[1] * (-s1[[1]]) + PauliMatrix[2] * (-s1[[2]]) +
        PauliMatrix[3] * (-s1[[3]])).(1
                                      0) +
     (0 1).(PauliMatrix[1] * a[[1]] + PauliMatrix[2] * a[[2]] + PauliMatrix[3] * a[[3]]).
      (PauliMatrix[1] * (-s1[[1]]) + PauliMatrix[2] * (-s1[[2]]) +
        PauliMatrix[3] * (-s1[[3]])).(0
                                      1))];

 aliceDet[[i]] = Extract[Flatten[Re[A]], 1];
 If[Abs[b.s] < z, B = {0 + 0. i},
  B = λ/2 ((1 0).(PauliMatrix[1] * s2[[1]] + PauliMatrix[2] * s2[[2]] + PauliMatrix[3] * s2[[3]]).
      (PauliMatrix[1] * b[[1]] + PauliMatrix[2] * b[[2]] + PauliMatrix[3] * b[[3]]).(1
                                                                                     0) +
     (0 1).(PauliMatrix[1] * s2[[1]] + PauliMatrix[2] * s2[[2]] + PauliMatrix[3] * s2[[3]]).
      (PauliMatrix[1] * b[[1]] + PauliMatrix[2] * b[[2]] + PauliMatrix[3] * b[[3]]).(0
                                                                                     1))];
 bobDet[[i]] = Extract[Flatten[Re[B]], 1],
 {i, trials}]
```

Statistical Analysis of Particle Data

In[845]:=
```
Do[
 If[(aliceDeg[[i]] * bobDeg[[i]]) > 0, theta = ArcCos[a1[[i]].b1[[i]]] * 180/π,
  theta = -ArcCos[a1[[i]].b1[[i]]] * 180/π + 360];
 θ = Round[theta];
 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[846]:=
```
pPP = 0; pPN = 0; pNP = 0; pNN = 0;
```

In[847]:=
```
mean = ConstantArray[0, trialDeg];
```

In[848]:=
```
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}]
```

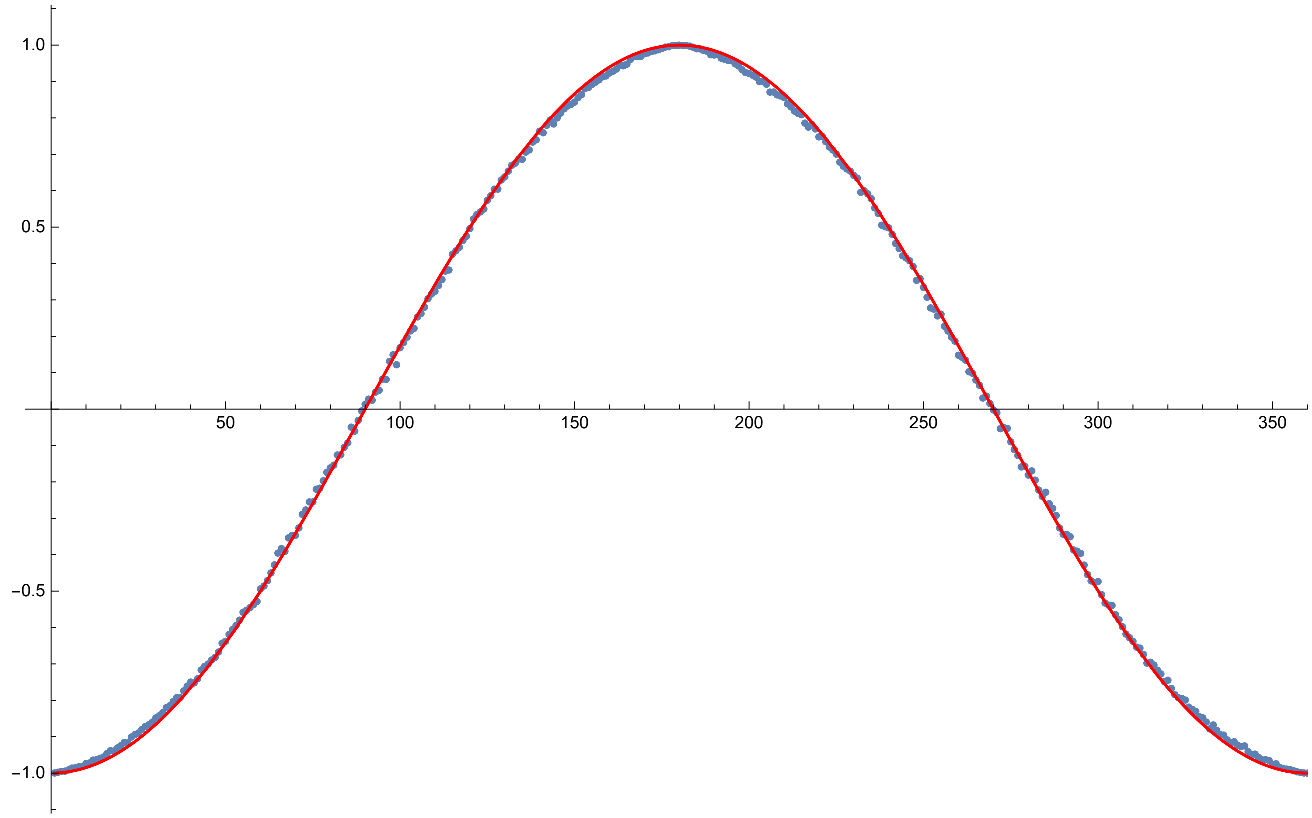
In[849]:=
```
simulation = ListPlot[mean, PlotMarkers → {Automatic, Tiny}];
```

In[850]:=
```
negcos = Plot[-Cos[x Degree], {x, 1, 361}, PlotStyle → {Red}];
```

Compare mean values with -Cosine Curve and compute averages

```
Show[simulation, negcos]
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}];
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]
```

Out[ ]=



AveA = −0.0006222

AveB = −0.0007768

P (A+) = 0.499624

P (B+) = 0.499538

Out[ ]= 0.249707

Out[ ]= 0.250575

Out[ ]= 0.249629

Out[ ]= 0.249595