Hybrid of two simulations by John Reed. Further modified by Fred Diether Feb. 2021 to work with 3D vectors, quaternions, and completely local realistic.

Set Run Time Parameters, Initialize Arrays and Tables

In[148]:=
```
<< Quaternions`;
β0 = Quaternion[1, 0, 0, 0];
β1 = Quaternion[0, 1, 0, 0];
β2 = Quaternion[0, 0, 1, 0];
β3 = Quaternion[0, 0, 0, 1];
Qcoordinates = {β1, β2, β3};
m = 1000000;
ABArray = Table[{0, 0, 0}, m];
```

Generate Particle Data

In[156]:=
```
For[nn = 1, nn ≤ m, nn += 1,
 {vectorA = RandomPoint[Sphere[]];
   vectorB = RandomPoint[Sphere[]];
   vectorS = RandomPoint[Sphere[]];
   λ = RandomReal[{0, 0.5}];
   Da = vectorA.Qcoordinates;
   Db = vectorB.Qcoordinates; (*Convert to quaternion coordinates*)
   Ls = vectorS.Qcoordinates; (*singlet spin quaternion*)
   ar = RandomPoint[Sphere[]];
   Dar = ar.Qcoordinates;
   br = RandomPoint[Sphere[]];
   Dbr = br.Qcoordinates;
   If[Abs[vectorA.vectorS] < λ, A = Sign[Re[-Dar ** Ls]], A = Sign[Re[- Da ** Ls]]];
   If[Abs[vectorB.vectorS] < λ, B = Sign[Re[Ls ** Dbr]], B = Sign[Re[Ls ** Db]]];
   If[Abs[vectorA.vectorS] < λ, aa = ar, aa = vectorA];
   If[Abs[vectorB.vectorS] < λ, bb = br, bb = vectorB];
   ϕA = ArcTan[vectorA[[2]], vectorA[[1]]];
   ϕB = ArcTan[vectorB[[1]], vectorB[[2]]];
   ϕAr = ArcTan[ar[[2]], ar[[1]]];
   ϕBr = ArcTan[br[[1]], br[[2]]];
   If[ϕA * ϕB > 0, angle = ArcCos[aa.bb] / Degree, angle = (-ArcCos[aa.bb]) / Degree + 360];
   ABArray[[nn]] = {Round[angle], A, B};};]
```

Statistical Analysis of Particle Data

In[157]:=
```
trials = m;
trialDeg = 360;
theta = ConstantArray[0, trials];
A1 = ConstantArray[0, trials];
B1 = ConstantArray[0, trials];
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];
theta = ABArray[[All, 1]]; A1 = ABArray[[All, 2]]; B1 = ABArray[[All, 3]];
Do[θ = theta[[j]];
 aliceD = A1[[j]]; bobD = B1[[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}]
```

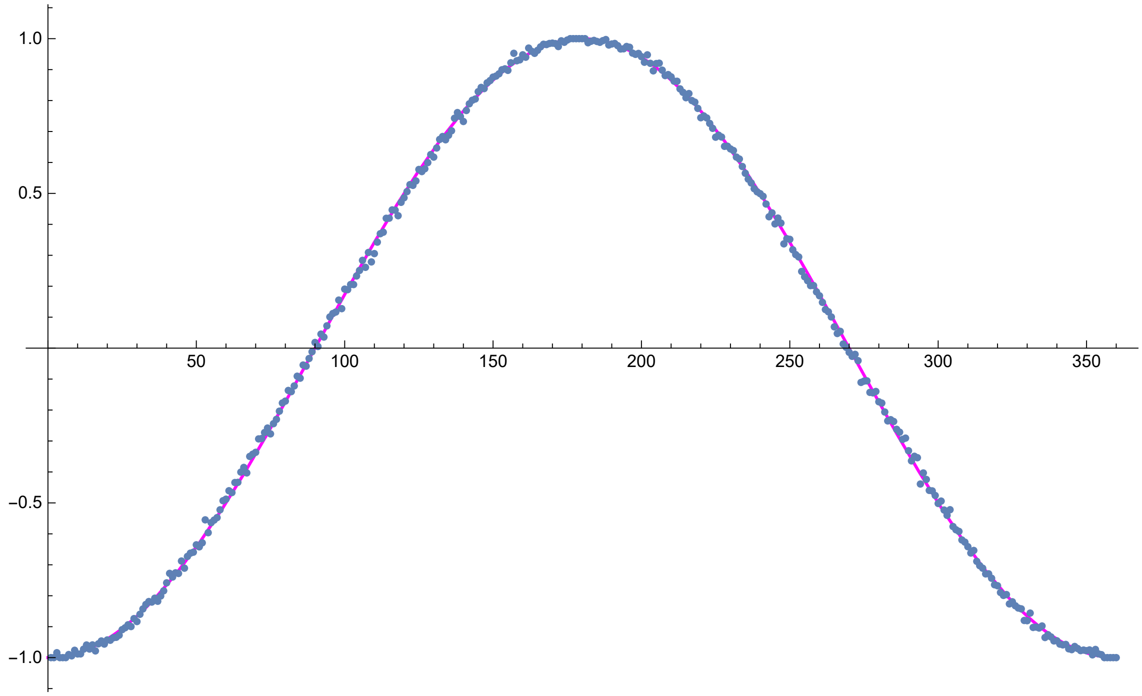
Calculate Mean Values

In[172]:=
```
pPP = 0; pPN = 0; pNP = 0; pNN = 0;
mean = ConstantArray[0, trialDeg];
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}]
simulation2 = ListPlot[mean, PlotMarkers → {Automatic, Tiny}];
negcos = Plot[-Cos[x Degree], {x, 0, 360}, PlotStyle → {Magenta}];
```

Plot comparing mean values with -Cosine Curve and compute averages

In[177]:=
```
Show[negcos, simulation2]
```

Out[177]=



In[178]:=
```
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[[68]]] + N[mean[[45]]] - N[mean[[135]]]];
Print["CHSH= ", CHSH]
```

AveA = −0.000266

AveB = −0.000702

P(A+) = 0.499867

P(B+) = 0.499649

Out[190]= 999 984

Out[191]= 0.249578

Out[192]= 0.250062

Out[193]= 0.250289

Out[194]= 0.250071

CHSH= 2.66134