**Simulation Based on Michel Fodje's epr-simple simulation translated from Python to Mathematica by John Reed 13 Nov 2013 and Quaternions**
**Modified by Fred Diether for Completely Local-Realistic Oct. 2021**
**Includes Joy's $S^3$ Quaternion Model. With 3D Vectors!**

Load Quaternion Package, Set Run Time Parameters, Initialize Arrays and Tables

In[92]:=
```
<< 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;
trialDeg = 721;
Θ3 = ConstantArray[0, m];
As1 = ConstantArray[0, m];
Bs2 = ConstantArray[0, m];
Ls1 = ConstantArray[0, m];
Ls2 = ConstantArray[0, m];
λ1 = ConstantArray[0, m];
λ2 = ConstantArray[0, m];
λ3 = ConstantArray[0, m];
qa = ConstantArray[0, m];
aq1 = ConstantArray[0, m];
qa1 = ConstantArray[0, m];
qa2 = ConstantArray[0, m];
qb = ConstantArray[0, m];
bq1 = ConstantArray[0, m];
qb1 = ConstantArray[0, m];
aa1 = ConstantArray[0, m];
bb1 = ConstantArray[0, m];
aa2 = ConstantArray[0, m];
bb2 = ConstantArray[0, m];
outA1 = Table[{0, 0, 0}, m];
outA2 = Table[{0, 0, 0}, m];
outB1 = Table[{0, 0, 0}, m];
outB2 = Table[{0, 0, 0}, m];
plotq = Table[{0, 0}, m];
a1 = ConstantArray[0, m];
b1 = ConstantArray[0, m];
A = ConstantArray[0, m];
B = ConstantArray[0, m];
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];
ϕ = 3; β = 0.25; ξ = 0.97;  (*Adustable parameters for fine tuning*)
```

## Generating Particle Data with Three Independent Do-Loops

In[137]:=
```
Do[θ = RandomPoint[Sphere[]];  (*Singlet 3D vector*)  (*Hidden Variable*)
 θ2 = Normalize[{θ[[1]], θ[[2]], 0}];
 θ3[[i]] = θ;
 θ1 = ToSphericalCoordinates[θ][[3]] * 180 / π;
 λ1[[i]] = β (Cos[θ1 / ϕ]^2);
 λ2[[i]] = ξ (Cos[θ1 / ϕ]^2);
 λ3[[i]] = Sign[θ1];
 As1[[i]] = θ2.Qcoordinates;
 Bs2[[i]] = -θ2.Qcoordinates;
 Ls1[[i]] = λ3[[i]] * θ.Qcoordinates;  (*Convert to quaternion coordinates*)
 Ls2[[i]] = -λ3[[i]] * θ.Qcoordinates, {i, m}]
```

In[138]:=
```
Do[a = RandomPoint[Sphere[]];  (*Detector vector angle*)
 aa = Normalize[{a[[1]], a[[2]], 0}];
 aa1[[i]] = a;
 aa2[[i]] = aa;
 Da = a.Qcoordinates;  (*Convert to quaternion coordinates*)
 Da1 = aa.Qcoordinates;
 qa = Da ** Ls1[[i]];
 qa2[[i]] = qa;
 qa1[[i]] = Da1 ** As1[[i]];
 aq = -Da ** Ls1[[i]];
 aq1[[i]] = aq;
 If[Abs[Re[qa]] > λ1[[i]], Aa1 = Sign[Re[qa]], Aa1 = Re[aq]];
 outA1[[i]] = {a, Aa1, i}, {i, m}]
Do[If[Abs[Re[qa1[[i]]]] < λ2[[i]], Aa2 = -Sign[qa1[[i]][[4]]], Aa2 = aq1[[i]][[4]]];
 outA2[[i]] = {aa2[[i]], Aa2, i + m}, {i, m}]
outA = Catenate[{outA1, outA2}];
```

In[141]:=
```
Do[b = RandomPoint[Sphere[]];  (*Detector vector angle*)
 bb = Normalize[{b[[1]], b[[2]], 0}];
 bb1[[i]] = b;
 bb2[[i]] = bb;
 Db = b.Qcoordinates;  (*Convert to quaternion coordinates*)
 Db1 = bb.Qcoordinates;
 qb = Ls2[[i]] ** Db;
 qb1[[i]] = Bs2[[i]] ** Db1;
 bq = -Ls2[[i]] ** Db;
 bq1[[i]] = bq;
 If[Abs[Re[qb]] > λ1[[i]], Bb1 = Sign[Re[qb]], Bb1 = Re[bq]];
 outB1[[i]] = {b, Bb1, i}, {i, m}]
Do[If[Abs[Re[qb1[[i]]]] < λ2[[i]], Bb2 = Sign[qb1[[i]][[4]]], Bb2 = bq1[[i]][[4]]];
 outB2[[i]] = {bb2[[i]], Bb2, i + m}, {i, m}]
outB = Catenate[{outB1, outB2}];
```

## Statistical Analysis of the Particle Data Received from Alice and Bob

```
m2 = 2 m;
theta = ConstantArray[0, m2];
th1 = ConstantArray[0, m2];
th2 = ConstantArray[0, m2];
th3 = ConstantArray[0, m2];
th4 = ConstantArray[0, m2];
a1 = outA[[All, 1]]; A = outA[[All, 2]]; b1 = outB[[All, 1]]; B = outB[[All, 2]];
Do[ϕA = ArcTan[a1[[i]][[1]], a1[[i]][[2]]] / 50;
 ϕB = ArcTan[b1[[i]][[2]], b1[[i]][[1]]] / 50;
 If[ϕA * ϕB > 0, th1[[i]] = ArcCos[a1[[i]].b1[[i]]], th1[[i]] = -ArcCos[a1[[i]].b1[[i]]]];
 theta[[i]] = Round[th1[[i]] * 180 / π + 181];
 th3[[i]] = Round[th1[[i]] * 180 / π + 181];
 th4[[i]] = Round[th1[[i]] * 180 / π + 181 + 360];
 th = th3[[i]];
 th5 = th4[[i]];
 aliceD = A[[i]]; bobD = B[[i]];
 If[aliceD == 1, nAP[[th]] ++];
 If[bobD == 1, nBP[[th]] ++];
 If[aliceD == -1, nAN[[th]] ++];
 If[bobD == -1, nBN[[th]] ++];
 If[aliceD == 1 && bobD == 1, nPP[[th]] ++];
 If[aliceD == 1 && bobD == -1, nPN[[th]] ++];
 If[aliceD == -1 && bobD == 1, nNP[[th]] ++];
 If[aliceD == -1 && bobD == -1, nNN[[th]] ++];
 If[aliceD == 1 && bobD == 1, nPP[[th5]] ++];
 If[aliceD == 1 && bobD == -1, nPN[[th5]] ++];
 If[aliceD == -1 && bobD == 1, nNP[[th5]] ++];
 If[aliceD == -1 && bobD == -1, nNN[[th5]] ++], {i, m2}]
```

## Calculating Mean Values of A, B, and AB, and Plotting the Results

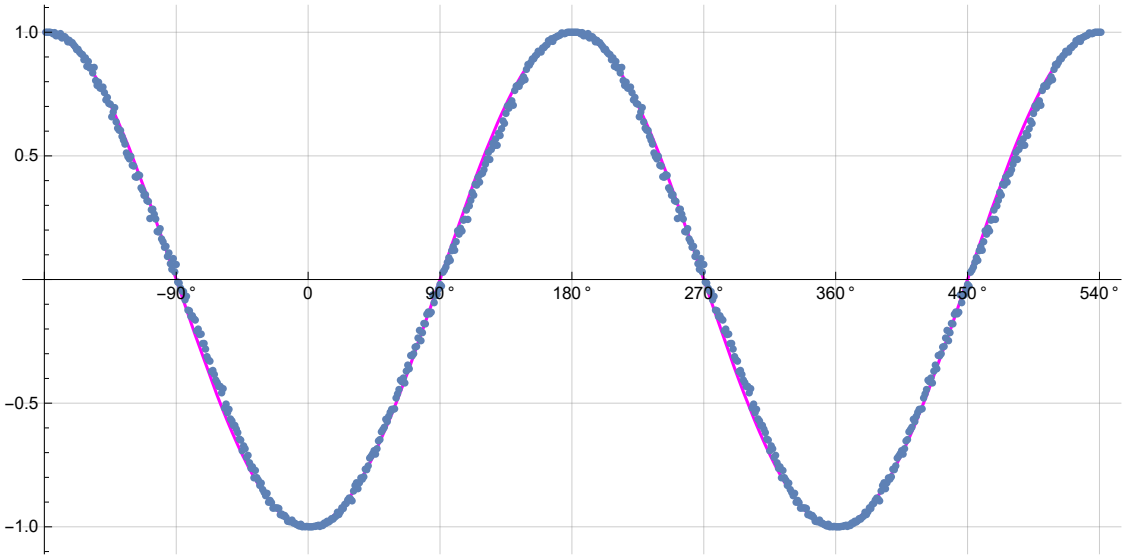
In[152]:=
```
mean = ConstantArray[0, trialDeg];
sum1 = ConstantArray[0, trialDeg];
sum2 = ConstantArray[0, trialDeg];
Do[sum1[[i]] = (nPP[[i]] + nNN[[i]] - nPN[[i]] - nNP[[i]]);
 sum2[[i]] = nPP[[i]] + nPN[[i]] + nNP[[i]] + nNN[[i]] + 0.0000001;
mean[[i]] = sum1[[i]] / sum2[[i]], {i, trialDeg}]
```

In[156]:=
```
simulation = ListPlot[mean, PlotMarkers → {Automatic, Tiny}];
negcos = Plot[Cos[x Degree], {x, 0, 720}, PlotStyle → {Magenta}, AspectRatio → 8 / 16,
   Ticks → {{{0, -180 °}, {90, -90 °}, {180, -0 °}, {270, 90 °}, {360, 180 °}, {450, 270 °},
      {540, 360 °}, {630, 450 °}, {720, 540 °}}, Automatic}, GridLines → Automatic];
```

## Comparing Mean Values with -Cosine Function and Computing Averages

In[158]:=
```
Show[negcos, simulation]
```

Out[158]=




In[159]:=
```
AveA = N[Sum[A[[i]], {i, m}] / m];
AveB = N[Sum[B[[i]], {i, m}] / m];
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}];
Print["Total Events = ", totAB / 2]
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];
Print["Ave ++ = ", PP]
Print["Ave -- = ", NN]
Print["Ave +- = ", PN]
Print["Ave -+ = ", NP]
CHSH = Abs[N[mean[[23]]] - N[mean[[135]]] + N[mean[[67]]] + N[mean[[45]]]];
Print["Approx. CHSH = ", CHSH]
```

```
AveA = 0.00104295

AveB = -0.000513863

P(A+)= 0.500431

P(B+)= 0.499679

Total Events = 986 379

Ave ++ = 0.25015

Ave -- = 0.249675

Ave +- = 0.250322

Ave -+ = 0.249853

Approx. CHSH = 2.69424
```