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 Sep 2021
Some parts by Bill Nelson. Includes Joy's S^3 Quaternion Model.

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

In[2367]:=
```
<< 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;
Ls1 = ConstantArray[0, m];
Ls2 = ConstantArray[0, m];
λ1 = ConstantArray[0, m];
λ2 = ConstantArray[0, m];
qa = ConstantArray[0, m];
aq = ConstantArray[0, m];
aq1 = ConstantArray[0, m];
qa1 = ConstantArray[0, m];
qb = ConstantArray[0, m];
aq = ConstantArray[0, m];
bq1 = ConstantArray[0, m];
qb1 = ConstantArray[0, m];
aa1 = ConstantArray[0, m];
bb1 = ConstantArray[0, m];
outA1 = Table[{0, 0, 0, 0}, m];
outA2 = Table[{0, 0, 0, 0}, m];
outB1 = Table[{0, 0, 0, 0}, m];
outB2 = Table[{0, 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 = 0.3; ξ = 0.921;  (*Adustable parameters for fine tuning*)
```

Generating Particle Data with Three Independent Do-Loops

In[2407]:=
```
Do[θ = RandomReal[{-179, 180}]; (*Singlet vector angle*) (*Hidden Variable*)
 λ1[[i]] = β (Cos[θ/3]^2);
 λ2[[i]] = ξ (Cos[θ/3]^2);
 θθ = N[Flatten[{FromPolarCoordinates[{1, θ * π/180}], 0}]];
 Ls1[[i]] = θθ.Qcoordinates;
 Ls2[[i]] = -θθ.Qcoordinates, {i, m}]
```

In[2408]:=
```
Do[a = RandomInteger[{-179, 180}]; (*Detector vector angle 1 degree increments*)
 aa1[[i]] = a;
 aa = N[Flatten[{FromPolarCoordinates[{1, a * π/180}], 0}]];
 Da = aa.Qcoordinates; (*Convert to quaternion coordinates*)
 qa = Da ** Ls1[[i]];
 qa1[[i]] = qa;
 aq = - (-Da ** Ls1[[i]]);
 aq1[[i]] = aq;
 If[Abs[Re[qa]] > λ1[[i]], Aa1 = Sign[Re[qa]], Aa1 = { }];
 outA1[[i]] = {a, Aa1, i, qa}, {i, m}]
Do[If[Abs[Re[aq1[[i]]]] < λ2[[i]], Aa2 = -Sign[aq1[[i]][[4]]], Aa2 = { }];
 outA2[[i]] = {aa1[[i]], Aa2, i + m, qa1[[i]]}, {i, m}]
outA = Catenate[{outA1, outA2}];
Length[outA2]
```

Out[2411]= 1 000 000

In[2412]:=
```
Do[b = RandomInteger[{-179, 180}]; (*Detector vector angle 1 degree increments*)
 bb1[[i]] = b;
 bb = N[Flatten[{FromPolarCoordinates[{1, b * π/180}], 0}]];
 Db = bb.Qcoordinates; (*Convert to quaternion coordinates*)
 qb = Ls2[[i]] ** Db;
 qb1[[i]] = qb;
 bq = - (-Ls2[[i]] ** Db);
 bq1[[i]] = bq;
 If[Abs[Re[qb]] > λ1[[i]], Bb1 = Sign[Re[qb]], Bb1 = { }];
 outB1[[i]] = {b, Bb1, i, qb}, {i, m}]
Do[If[Abs[Re[bq1[[i]]]] < λ2[[i]], Bb2 = Sign[bq1[[i]][[4]]], Bb2 = { }];
 outB2[[i]] = {bb1[[i]], Bb2, i + m, qb1[[i]]}, {i, m}]
outB = Catenate[{outB1, outB2}];
Length[outB2]
```

Out[2415]= 1 000 000

Statistical Analysis of the Particle Data Received from Alice and Bob

In[2416]:=
```
m2 = 2 * m;
theta = ConstantArray[0, m2];
a1 = outA[[All, 1]]; A = outA[[All, 2]]; b1 = outB[[All, 1]]; B = outB[[All, 2]];
Do[th = a1[[i]] - b1[[i]] + 361;
 (*All angles are shifted by 361 degrees since θ1 is an index*)
 theta[[i]] = th;
 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]]++], {i, m2}]
```

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

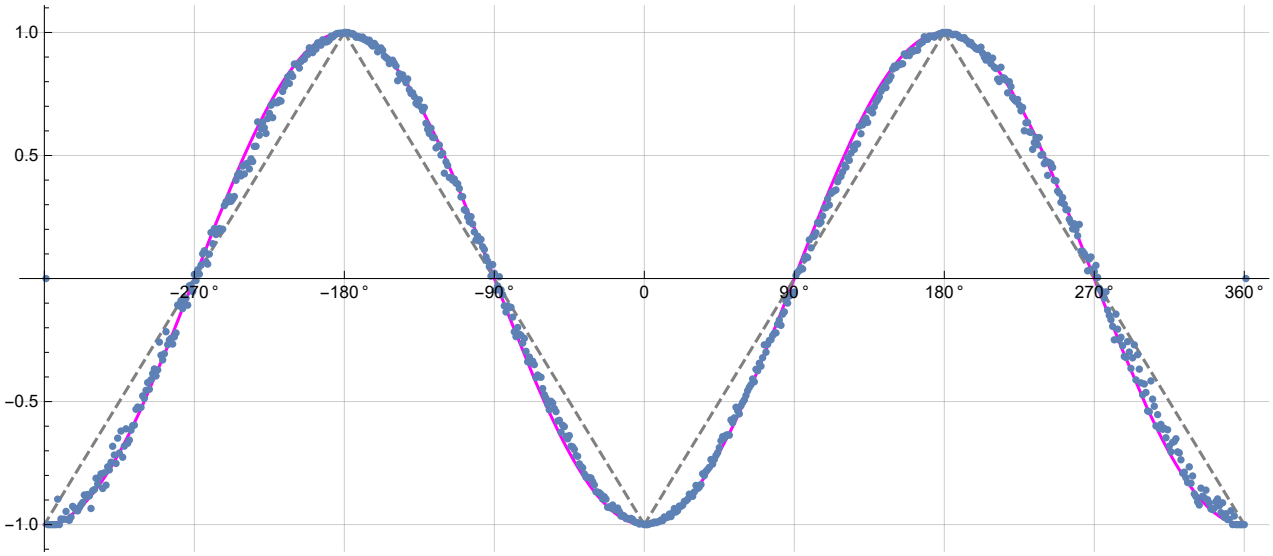
In[2420]:=
```
mean = ConstantArray[0, trialDeg];
sum1 = ConstantArray[0, trialDeg];
sum2 = ConstantArray[0, trialDeg];
Do[sum1[[i]] = (nPP[[i]] + nNN[[i]] - nPN[[i]] - nNP[[i]]);
 sum2[[i]] = nPP[[i]] + nPN[[i]] + nNP[[i]] + nNN[[i]] + 0.0000001;
mean[[i]] = sum1[[i]] / sum2[[i]], {i, trialDeg}]
```

In[2424]:=
```
simulation = ListPlot[mean, PlotMarkers → {Automatic, Tiny}];
negcos = Plot[-Cos[x Degree], {x, 0, 720}, PlotStyle → {Magenta}, AspectRatio → 7 / 16,
   Ticks → {{{0, -360 °}, {90, -270 °}, {180, -180 °}, {270, -90 °}, {360, 0 °}, {450, 90 °},
      {540, 180 °}, {630, 270 °}, {720, 360 °}}, Automatic}, GridLines → Automatic];
p1 = Plot[-1 + 2 x Degree / π, {x, 0, 180}, PlotStyle → {Gray, Dashed}];
p2 = Plot[3 - 2 x Degree / π, {x, 180, 360}, PlotStyle → {Gray, Dashed}];
p3 = Plot[-5 + 2 x Degree / π, {x, 360, 540}, PlotStyle → {Gray, Dashed}];
p4 = Plot[7 - 2 x Degree / π, {x, 540, 720}, PlotStyle → {Gray, Dashed}];
```

Comparing Mean Values with -Cosine Function and Computing Averages

In[2430]:=
```
Show[negcos, p1, p2, p3, p4, simulation]
```

Out[2430]=



In[2431]:=

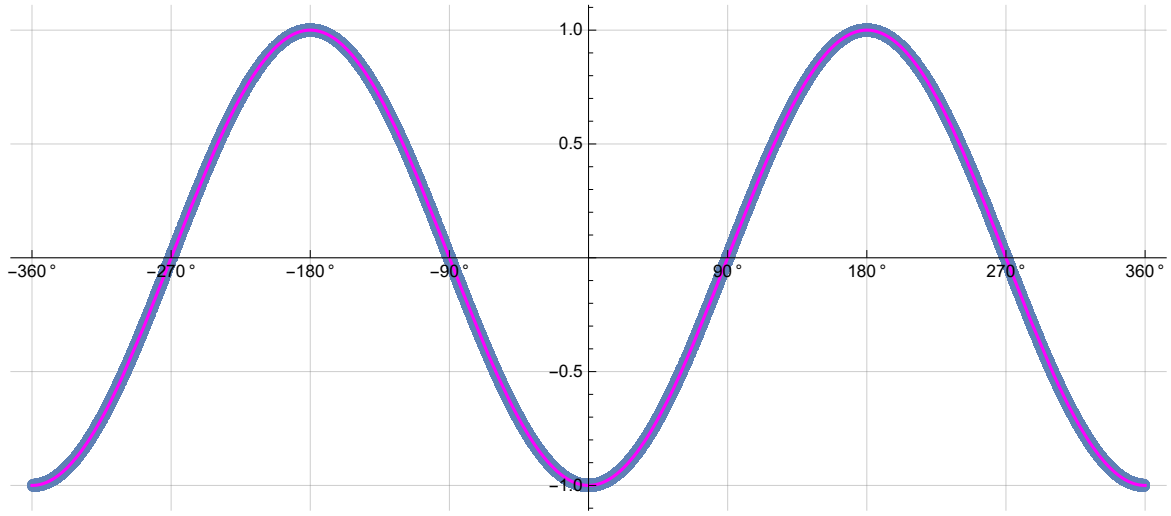
```
A1 = DeleteCases[A, {}];
m3 = Length[A1];
B1 = DeleteCases[B, {}];
m4 = Length[B1];
AveA = N[Sum[A1[[i]], {i, m3}] / m3];
AveB = N[Sum[B1[[i]], {i, m4}] / m4];
Print["AveA = ", AveA];
Print["AveB = ", AveB];
PAP = N[Sum[nAP[[i]], {i, trialDeg}]];
PBP = N[Sum[nBP[[i]], {i, trialDeg}]];
PAN = N[Sum[nAN[[i]], {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]
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[[315]]] - N[mean[[225]]] + N[mean[[405]]] + N[mean[[45]]]];
Print["Approx. CHSH = ", CHSH]
```

```
AveA = 0.000721889

AveB = -0.000484366

P(A+)= 0.500361

P(B+)= 0.499758

Total Events = 998767

Ave ++ = 0.249562

Ave -- = 0.249807

Ave +- = 0.250687

Ave -+ = 0.249944

Approx. CHSH = 2.76107
```

Product Calculation

In[2459]:=
```
q = 0;
q2a = outA[[All, 4]];
q2b = outB[[All, 4]];
plotq = Table[{0, 0}, m];
angle = ConstantArray[0, m];
Do[If[λ2[[i]] == 1, q = q2a[[i]] ** q2b[[i]], q = q2b[[i]] ** q2a[[i]]];
 angle = theta[[i]] - 361;
 plotq[[i]] = {angle, Re[q]}, {i, m}]
sim = ListPlot[plotq, PlotMarkers → {Automatic, Small}, AspectRatio → 7 / 16,
   Ticks → {{{-360, -360 °}, {-270, -270 °}, {-180, -180 °}, {-90, -90 °}, {0, 0 °}, {90, 90 °},
      {180, 180 °}, {270, 270 °}, {360, 360 °}}, Automatic}, GridLines → Automatic];
negcos1 = Plot[-Cos[x Degree], {x, -360, 360}, PlotStyle → {Magenta}];
Show[sim, negcos1]
```

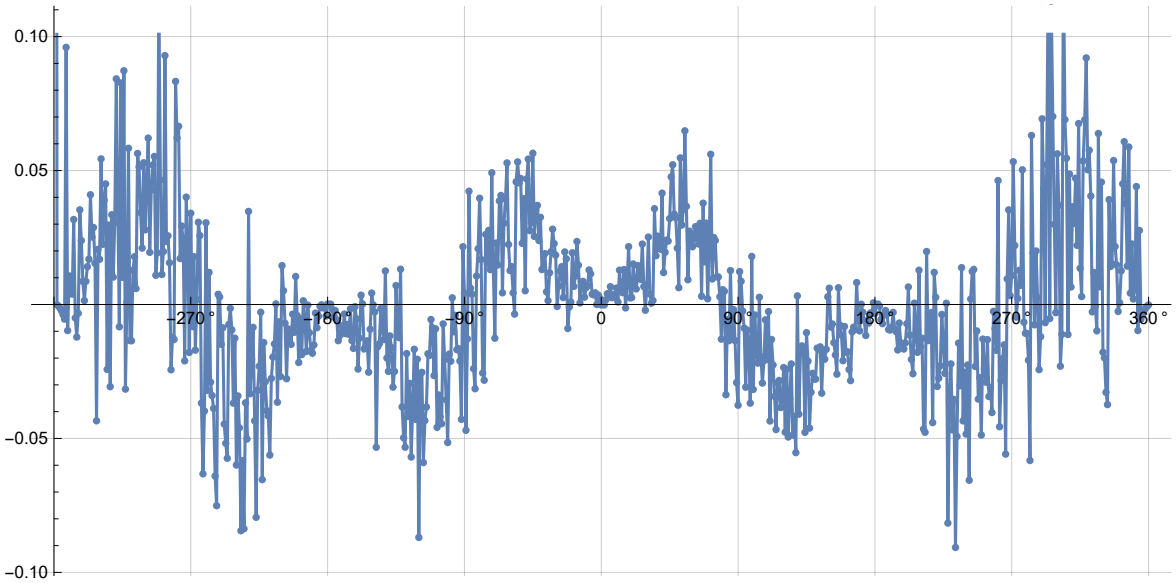
Out[2467]=



Blue is the correlation data and magenta is the -cosine curve for an exact match.

In[2468]:=
```
dev1 = ConstantArray[2, 720];
dev2 = ConstantArray[2, 720];
dev3 = ConstantArray[2, 720];
Do[dev1 = mean[[i]]; dev2[[i]] = {dev1, i}, {i, 720}]
devang = dev2[[All, 2]] - 361;
Do[dev3[[i]] = mean[[i]] + Cos[devang[[i]] Degree], {i, 720}]
ListPlot[N[dev3], PlotMarkers → {Automatic, Tiny}, Joined → True, AspectRatio → 1 / 2,
 Ticks → {{{0, -360 °}, {90, -270 °}, {180, -180 °}, {270, -90 °}, {360, 0 °}, {450, 90 °},
    {540, 180 °}, {630, 270 °}, {720, 360 °}}, Automatic}, GridLines → Automatic]
```

Out[2474]=



In[2475]:=
```
N[Mean[Abs[dev3]]]
N[Mean[dev3]]
```

Out[2475]= 0.0241046

Out[2476]= 0.00156817