Simulation Based on Joy Christian's updated 3-Sphere model and Michel Fodje's epr-simple simulation translated from Python to Mathematica by John Reed 13 Nov 2013 plus some Quaternion Parts and using Reed's trial number matching code. Modified, Created by Fred Diether for Completely Local-Realistic Feb. 2022

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

In[1]:=
```
<< 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};
Qcoordinates2 = {β0, β1, β2, β3};
m = 5000000; (*Number of events to perform*)
ss = ConstantArray[0, m];
λ = ConstantArray[0, m];
Ls1 = ConstantArray[0, m];
Ls2 = ConstantArray[0, m];
a1 = ConstantArray[0, m];
b1 = ConstantArray[0, m];
aa1 = ConstantArray[0, m];
bb1 = ConstantArray[0, m];
qa1 = ConstantArray[0, m];
qb1 = ConstantArray[0, m];
outqA = ConstantArray[0, m];
outqB = ConstantArray[0, m];
λA = ConstantArray[0, m];
λB = ConstantArray[0, m];
λ1 = ConstantArray[0, m];
λ2 = ConstantArray[0, m];
m2 = 20000; (*Events to perform for product calculation and CHSH*)
r0 = ConstantArray[0, m2];
r1 = ConstantArray[0, m2];
r2 = ConstantArray[0, m2];
QAB = ConstantArray[0, m2];
plotq = Table[{0, 0}, m2];
ssca = ConstantArray[0, m];
sscb = ConstantArray[0, m];
nPP = ConstantArray[0, 720];
nNN = ConstantArray[0, 720];
nPN = ConstantArray[0, 720];
nNP = ConstantArray[0, 720];
nAP = ConstantArray[0, 720];
nBP = ConstantArray[0, 720];
ϕ = 3; β = 0.24; ξ = -15;(*Adjustable parameters*)
```

Generating Particle Data with Three Independent Do-Loops and Implement Hidden Variable Mechanisms

We note here that the Limit[x, x → Sign[x]] = Sign[x] so that we just use the sign function in the A and B Do-loops instead of the limits.

In[40]:=
```
Do[s = RandomPoint[Sphere[]];  (*Singlet 3D vector; Hidden Variable*)
 Θ1 = ToSphericalCoordinates[s][[3]] * 180 / π;
 ss[[k]] = Θ1;
 λ[[k]] = β (Cos[Θ1 / ϕ]^2);  (*Hidden variable mechanism*)
 Ls1[[k]] = s.Qcoordinates;  (*Convert to quaternion; A particle spin*)
 Ls2[[k]] = -s.Qcoordinates, {k, m}]  (*B particle spin plus conservation of angular momentum*)
```

In[41]:=
```
Do[a = RandomInteger[{-179, 180}];  (*Detector 2D vector angle 1 degree increments*)
 a1[[k]] = a;
 aa = N[Flatten[{FromPolarCoordinates[{1, a * π / 180}], 0.000001}]];
 aa1[[k]] = aa;
 Da = aa.Qcoordinates;  (*Convert to quaternion; A detector*)
 qa = Da ** Ls1[[k]];  (*Detector - particle interaction*)
 qa1[[k]] = qa;
 If[Abs[Re[qa]] > λ[[k]], qA = Sign[Re[qa]], qA = Sign[Sin[(a - ss[[k]] + ξ) Degree]]];
 outqA[[k]] = qA;
 λA[[k]] = Sign[Sin[(a - ss[[k]] + ξ) Degree]];  (*Hidden variable mechanism*)
 If[Abs[Re[qa]] > λ[[k]], λ1[[k]] = 0, λ1[[k]] = k], {k, m}]  (*Hidden variable mechanism*)
outqA3 = outqA;
```

In[43]:=
```
Do[b = RandomInteger[{-179, 180}];  (*Detector 2D vector angle 1 degree increments*)
 b1[[k]] = b;
 bb = N[Flatten[{FromPolarCoordinates[{1, b * π / 180}], 0.000001}]];
 bb1[[k]] = bb;
 Db = bb.Qcoordinates;  (*Convert to quaternion; B detector*)
 qb = Ls2[[k]] ** Db;  (*Detector - particle interaction*)
 qb1[[k]] = qb;
 If[Abs[Re[qb]] > λ[[k]], qB = Sign[Re[qb]], qB = -Sign[Sin[(b - ss[[k]] + ξ) Degree]]];
 outqB[[k]] = qB;
 λB[[k]] = -Sign[Sin[(b - ss[[k]] + ξ) Degree]];  (*Hidden variable mechanism*)
 If[Abs[Re[qb]] > λ[[k]], λ2[[k]] = 0, λ2[[k]] = k], {k, m}]  (*Hidden variable mechanism*)
outqB3 = outqB;
```

Verification of the 3-Sphere Model Product Calculation Prediction

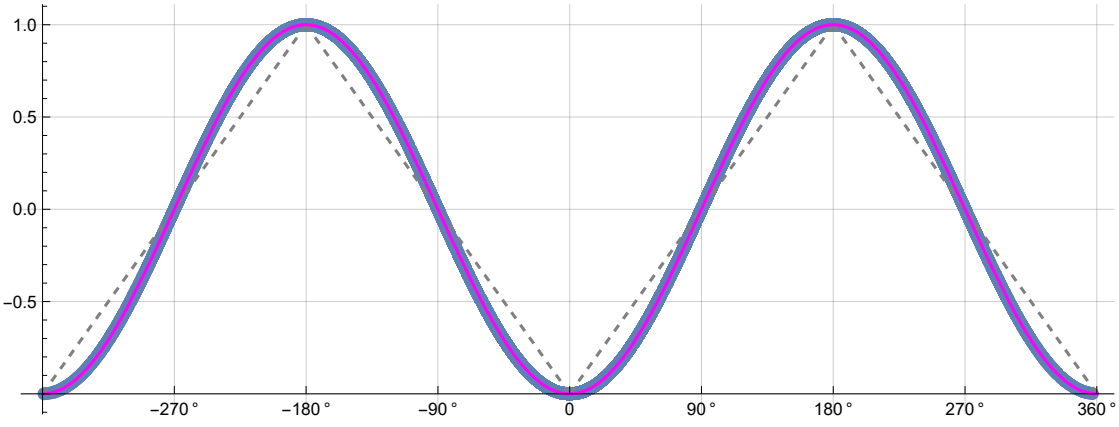
In[45]:=
```
(*qA=Re[Da**Limit[Ls1[[k]],Ls1[[k]]→Sign[Re[Da**Ls1[[k]]]]Da]];
qB=Re[Db**Limit[Ls2[[k]],Ls2[[k]]→Sign[Re[Db**Ls2[[k]]]]Db]];*)
(*The above two lines are moved to the independent A and B Do-loops
  and modified for proper calculation of the A and B outcomes.*)
Do[r1[[k]] = {qa1[[k]][[2]], qa1[[k]][[3]], qa1[[k]][[4]]};
 r2[[k]] = {qb1[[k]][[2]], qb1[[k]][[3]], qb1[[k]][[4]]};
 QAB[[k]] = Re[qa1[[k]]] * Re[qb1[[k]]] - r1[[k]].r2[[k]];
r0[[k]] = (Re[qa1[[k]]] Limit[Cross[s2, bb1[[k]]], s2 → Sign[Re[qb1[[k]]]] bb1[[k]]]
      + Re[qb1[[k]]] Limit[Cross[aa1[[k]], s1], s1 → Sign[Re[qa1[[k]]]] aa1[[k]]]
      - Cross[Limit[Cross[aa1[[k]], s1], s1 → Sign[Re[qa1[[k]]]] aa1[[k]]],
       Limit[Cross[s2, bb1[[k]]], s2 → Sign[Re[qb1[[k]]]] bb1[[k]]]) /
   (Sin[ArcCos[aa1[[k]].bb1[[k]]]]);
 q = {Re[QAB[[k]]], r0[[k]][[1]], r0[[k]][[2]], r0[[k]][[3]]}.Qcoordinates2;
 θ = a1[[k]] - b1[[k]];
 plotq[[k]] = {θ, Re[q]}, {k, m2}]
```

In[46]:=
```
Print["Typical q = ", q];
meanq = Mean[plotq[[All, 2]]];
Print["Imaginary part vanishes. meanq = ", meanq];
sim1 = ListPlot[plotq, PlotMarkers → {Automatic, Small}, AspectRatio → 3 / 8,
   Ticks → {{{90, 90 °}, {-90, -90 °}, {-180, -180 °}, {180, 180 °}, {0, 0 °},
       {-270, -270 °}, {270, 270 °}, {-360, -360 °}, {360, 360 °}}, Automatic},
   GridLines → Automatic, AxesOrigin → {-360, -1.0}];
p1 = Plot[-1 + 2 x1 Degree / π, {x1, 0, 180}, PlotStyle → {Gray, Dashed}];
p2 = Plot[3 - 2 x2 Degree / π, {x2, 180, 360}, PlotStyle → {Gray, Dashed}];
p3 = Plot[3 + 2 x3 Degree / π, {x3, -360, -180}, PlotStyle → {Gray, Dashed}];
p4 = Plot[-1 - 2 x4 Degree / π, {x4, -180, 0}, PlotStyle → {Gray, Dashed}];
negcos1 = Plot[-Cos[x Degree], {x, -360, 360}, PlotStyle → {Magenta}];
Show[sim1, p1, p2, p3, p4, negcos1]
```

```
Typical q = Quaternion[0.71934, 0., 0., 0.]

Imaginary part vanishes. meanq = -0.00152204
```

Out[55]=



Blue is the correlation data, magenta is the negative cosine curve for an exact match.

Statistical Analysis of the Particle Data Received from Alice and Bob

Spinorial Sign Change Corrections in A and B

For the spinorial sign change corrections we will need,

$$\mathbf{q}(\eta_{\mathbf{sn}} + \delta\,\pi,\, \mathbf{r}) = (-1)^{\delta}\,\mathbf{q}(\eta_{\mathbf{sn}},\, \mathbf{r}) \quad \text{for } \delta = 0, 1, 2, 3, \ldots$$

In[56]:=
```
Do[If[λ2[[k]] == k && outqA[[k]] ≠ λA[[k]], outqA3[[k]] = outqA[[k]] * -1];
 (*Spinorial sign change correction*)
 If[λ1[[k]] == k && outqB[[k]] ≠ λB[[k]], outqB3[[k]] = outqB[[k]] * -1];
 (*Spinorial sign change correction*)
 If[λ2[[k]] == k && outqA[[k]] ≠ λA[[k]], ssca[[k]] = 1, ssca[[k]] = 0];
 If[λ1[[k]] == k && outqB[[k]] ≠ λB[[k]], sscb[[k]] = 1, sscb[[k]] = 0], {k, m}]
A = outqA3;
B = outqB3;
PercentForm [N[Total[ssca] / m]] (*Percentage of spinorial sign changes.*)
PercentForm[N[Total[sscb] / m]]
```

Out[59]//PercentForm=
```
3.416%
```

Out[60]//PercentForm=
```
3.42%
```

In[61]:=
```
Do[θ2 = a1[[k]] - b1[[k]] + 360;  (*Angles shifted by 360 since θ2 is an index*)
 aliceD = A[[k]]; bobD = B[[k]];
 If[aliceD == 1, nAP[[θ2]]++];
 If[bobD == 1, nBP[[θ2]]++];
 If[aliceD == 1 && bobD == 1, nPP[[θ2]]++];
 If[aliceD == 1 && bobD == -1, nPN[[θ2]]++];
 If[aliceD == -1 && bobD == 1, nNP[[θ2]]++];
 If[aliceD == -1 && bobD == -1, nNN[[θ2]]++], {k, m}]
```

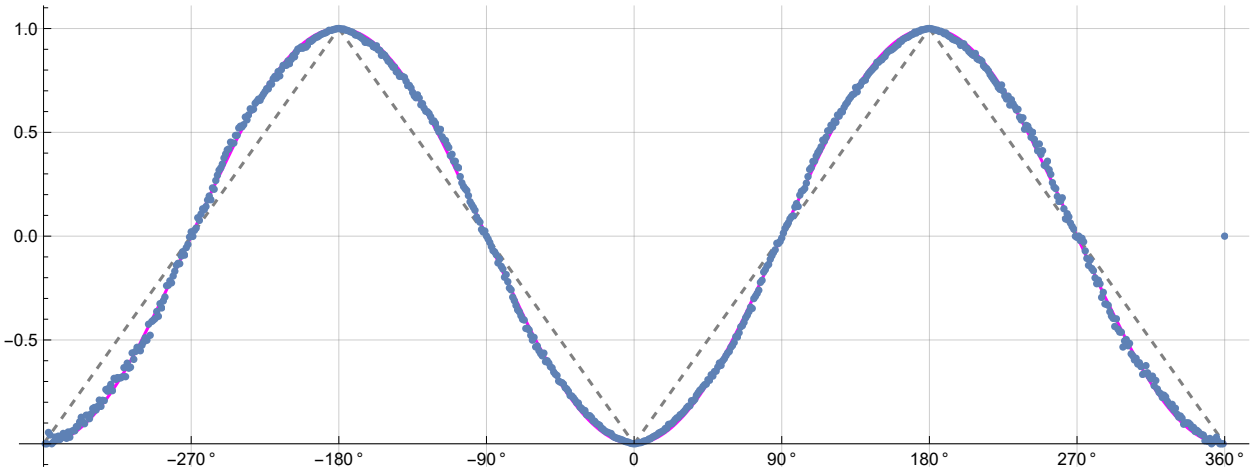
Calculating Mean Values of AB

In[62]:=
```
mean = ConstantArray[1, 720];
sum1 = ConstantArray[1, 720];
sum2 = ConstantArray[1, 720];
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, 1, 720}]
```

Plotting the Results Comparing Mean Values with -Cosine Curve

In[66]:=
```
sim2 = ListPlot[mean, PlotMarkers → {Automatic, Tiny}];
negcos2 = Plot[-Cos[x7 Degree], {x7, 0, 720}, PlotStyle → {Magenta}, AspectRatio → 3 / 8,
   Ticks → {{{0, -360 °}, {90, -270 °}, {180, -180 °}, {270, -90 °}, {360, 0 °}, {450, 90 °},
      {540, 180 °}, {630, 270 °}, {720, 360 °}}, Automatic}, GridLines → Automatic,
   AxesOrigin → {0, -1.0}];
p5 = Plot[-5 + 2 x5 Degree / π, {x5, 360, 540}, PlotStyle → {Gray, Dashed}];
p6 = Plot[7 - 2 x6 Degree / π, {x6, 540, 720}, PlotStyle → {Gray, Dashed}];
Show[negcos2, p1, p2, p5, p6, sim2]
```

Out[70]=



Computing Total Events Detected and Averages

In[71]:=
```
totAB = Total[nPP + nNN + nPN + nNP];
AveA = N[Total[A] / totAB];
AveB = N[Total[B] / totAB];
PAP = Total[nAP];
PBP = Total[nBP];
PA1 = N[PAP / totAB];
PB1 = N[PBP / totAB];
PP = N[Total[nPP] / totAB ];
NN = N[Total[nNN] / totAB ];
PN = N[Total[nPN] / totAB ];
NP = N[Total[nNP] / totAB ];
CHSH = Abs[N[mean[[315]] - N[mean[[225]] + N[mean[[405]] + N[mean[[45]]];
Print["Total Events Detected = ", totAB];
Print["AveA = ", AveA];
Print["AveB = ", AveB];
Print["P(A+) = ", PA1];
Print["P(B+) = ", PB1];
Print["P(++) = ", PP];
Print["P(--) = ", NN];
Print["P(+-) = ", PN];
Print["P(-+) = ", NP];
Print["Approx. CHSH = ", CHSH];
```

Total Events Detected = 5 000 000

AveA = −0.0002132

AveB = 0.0005564

P(A+) = 0.499893

P(B+) = 0.500278

P(++) = 0.250334

P(−−) = 0.250162

P(+−) = 0.24956

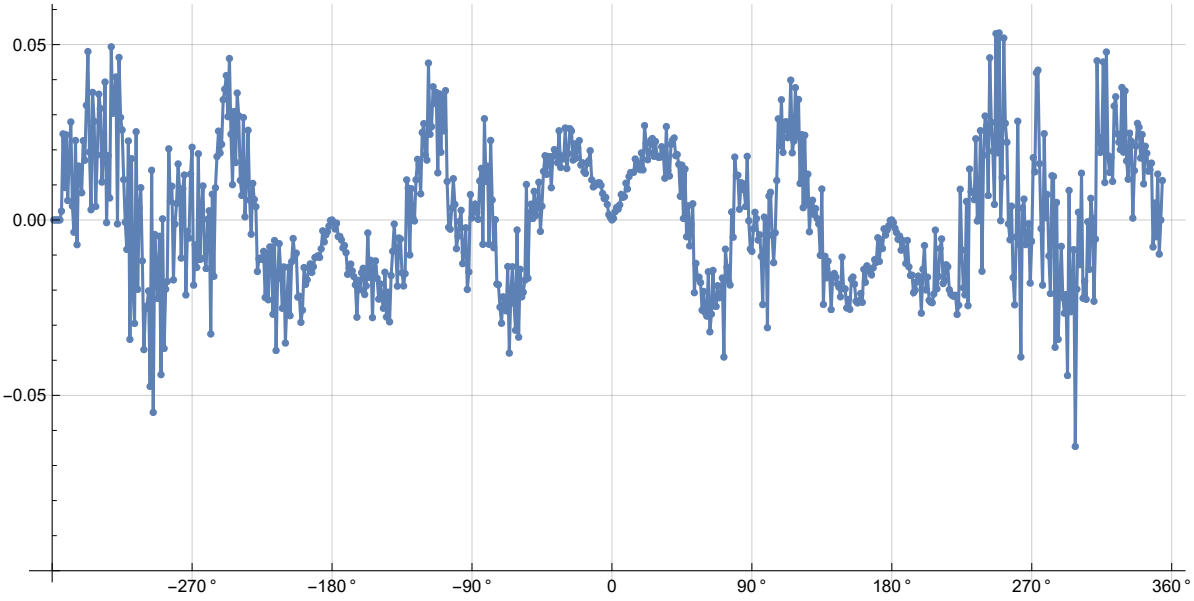
P(−+) = 0.249945

Approx. CHSH = 2.76811

Computing Deviation from -Cosine Data

In[93]:=
```
devang = ConstantArray[0, 714];
dev = ConstantArray[0, 714];
Do[devang[[i]] = i;
dev[[i]] = mean[[i]] + Cos[devang[[i]] Degree], {i, 6, 714}]
ListPlot[N[dev], PlotMarkers → {Automatic, Tiny}, Joined → True,
 AspectRatio → 8 / 16, Ticks → {{{0, -360 °}, {90, -270 °}, {180, -180 °}, {270, -90 °},
    {360, 0 °}, {450, 90 °}, {540, 180 °}, {630, 270 °}, {720, 360 °}}, Automatic},
 GridLines → Automatic, AxesOrigin → {0, -0.1}]
```

Out[96]=



In[127]:=
```
Mean[N[dev]] // PercentForm
Mean[N[Abs[dev]]]
```

Out[127]//PercentForm=

0.093%

Out[128]= 0.0157947

In[99]:=
```
-0.01349 × %; 0.0218666; (*deviations, 1 million events*)
```

CHSH Analysis

In[100]:=
```
λA2 = ConstantArray[0, m2];
λB2 = ConstantArray[0, m2];
outqA2 = ConstantArray[0, m2];
outqB2 = ConstantArray[0, m2];
a3 = ConstantArray[0, m2];
b3 = ConstantArray[0, m2];
λ3 = ConstantArray[0, m2];
λ4 = ConstantArray[0, m2];
```

Generating Particle Data with Independent Do-Loops

In[108]:=
```
Do[a2 = RandomChoice[{0, 90}];  (*Detector 2D vector angle*)
 a3[[k]] = a2;
 aa2 = N[Flatten[{FromPolarCoordinates[{1, a2 * π / 180}], 0}]];
 Da2 = aa2.Qcoordinates;  (*Convert to quaternion coordinates*)
 qa2 = Da2 ** Ls1[[k]];
 If[Abs[Re[qa2]] > λ[[k]], qA2 = Sign[Re[qa2]], qA2 = Sign[Sin[(a2 - ss[[k]] + ξ) Degree]]];
 outqA2[[k]] = qA2;
 λA2[[k]] = Sign[Sin[(a2 - ss[[k]] + ξ) Degree]];
 If[Abs[Re[qa2]] > λ[[k]], λ3[[k]] = 0, λ3[[k]] = k], {k, m2}]
outA2 = outqA2;
```

In[110]:=
```
Do[b2 = RandomChoice[{45, 135}];  (*Detector 2D vector angle*)
 b3[[k]] = b2;
 bb2 = N[Flatten[{FromPolarCoordinates[{1, b2 * π / 180}], 0}]];
 Db2 = bb2.Qcoordinates;  (*Convert to quaternion coordinates*)
 qb2 = Ls2[[k]] ** Db2;
 If[Abs[Re[qb2]] > λ[[k]], qB2 = Sign[Re[qb2]], qB2 = -Sign[Sin[(b2 - ss[[k]] + ξ) Degree]]];
 outqB2[[k]] = qB2;
 λB2[[k]] = -Sign[Sin[(b2 - ss[[k]] + ξ) Degree]];
 If[Abs[Re[qb2]] > λ[[k]], λ4[[k]] = 0, λ4[[k]] = k], {k, m2}]
outB2 = outqB2;
```

CHSH Analysis of the Particle Data Received from Alice and Bob

Spinorial Sign Change corrections in A and B

In[112]:=
```
Do[If[λ4[[k]] == k && outqA2[[k]] ≠ λA2[[k]], outA2[[k]] = outqA2[[k]] * -1];
 If[λ3[[k]] == k && outqB2[[k]] ≠ λB2[[k]], outB2[[k]] = outqB2[[k]] * -1], {k, m2}]
(*Spinorial sign changes for A and B*)
AA = outA2;
BB = outB2;
```

In[115]:=
```
nP1 = 0; nN1 = 0; nP2 = 0; nN2 = 0; nP3 = 0; nN3 = 0; nP4 = 0;  nN4 = 0;
Do[a1 = a3[[k]]; b1 = b3[[k]];
 aliceD = AA[[k]]; bobD = BB[[k]];
 If[(b1 == 45) && (a1 - b1 == -45) && aliceD * bobD == 1, nP1++];
 If[(b1 == 45) && (a1 - b1 == -45) && aliceD * bobD == -1, nN1++];
 If[(a1 - b1) == -135 && aliceD * bobD == 1, nP2++];
 If[(a1 - b1) == -135 && aliceD * bobD == -1, nN2++];
 If[(a1 - b1) == 45 && aliceD * bobD == 1, nP3++];
 If[(a1 - b1) == 45 && aliceD * bobD == -1, nN3++];
 If[a1 == 90 && (a1 - b1) == -45 && aliceD * bobD == 1, nP4++];
 If[a1 == 90 && (a1 - b1) == -45 && aliceD * bobD == -1, nN4++], {k, m2}]
E1 = N[(nP1 - nN1) / (nP1 + nN1)];
E2 = N[(nP2 - nN2) / (nP2 + nN2)];
E3 = N[(nP3 - nN3) / (nP3 + nN3)];
E4 = N[(nP4 - nN4) / (nP4 + nN4)];
tot1 = Total[nP1 + nP2 + nP3 + nP4 + nN1 + nN2 + nN3 + nN4];
CHSH = N[Abs[E1 - E2 + E3 + E4]];
Print["CHSH = ", CHSH]
Print["Total Events Detected = ", tot1]
```

```
CHSH = 2.78937
```

```
Total Events Detected = 20000
```

In[126]:=
```
(2.8323946013339243 + 2.8292076083106292 + 2.8335071953985898 +
   2.8260985341778437 + 2.8254732463559082 + 2.8043705677375788 + 2.800333098716658 +
   2.7828705966805343 + 2.8202429689740085 +2.7893712884149275) / 10
(*Average of 10 Runs of 20000 Trials Each to Machine Precision*)
```

Out[126]=
```
2.81439
```