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[106]:= << 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 = 2 000 000;
trialDeg = 721;
ss2 = ConstantArray[0, m];
Ls1 = ConstantArray[0, m];
Ls2 = ConstantArray[0, m];
λ1 = ConstantArray[0, m];
hA = ConstantArray[0, m];
hB = ConstantArray[0, m];
outAa = Table[{0, 0, 0, 0}, m];
outBb = Table[{0, 0, 0, 0}, m];
outA12 = Table[{0, 0, 0, 0}, m];
outA22 = Table[{0, 0, 0, 0}, m];
outB12 = Table[{0, 0, 0, 0}, m];
outB22 = Table[{0, 0, 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 = -15;  (*Adustable parameters for fine tuning*)
```

Generating Particle Data with Three Independent Do-Loops

```
In[139]:= Do[s = RandomPoint[Sphere[]];  (*Singlet 3D vector angle*)  (*Hidden Variable*)
  θ1 = ToSphericalCoordinates[s]〚3〛 * 180 / π;
  ss2〚i〛 = θ1;
  λ1〚i〛 = β (Cos[θ1/ϕ]^2);  (*Hidden variable mechanism*)
  Ls1〚i〛 = s.Qcoordinates;  (*Quaternion particle for the A side*)
  Ls2〚i〛 = -s.Qcoordinates, {i, m}]
 (*Quaternion particle for the B side*)(*Conservation of angular momentum*)

In[140]:= Do[a = RandomInteger[{-179, 180}];  (*Detector 2D vector angle 1 degree increments*)
  aa = N[Flatten[{FromPolarCoordinates[{1, a * π / 180}], 0}]];
  Da = aa.Qcoordinates;  (*Convert to quaternion coordinates*)
  qa = Da ** Ls1〚i〛;
  If[Abs[Re[qa]] > λ1〚i〛, Aa = Sign[Re[qa]], Aa = Sign[Sin[(a - ss2〚i〛 + ξ) Degree]]];
  A0 = Sign[Sin[(a - ss2〚i〛 + ξ) Degree]];
  outAa〚i〛 = {a, Aa, i, A0};
  If[Abs[Re[qa]] > λ1〚i〛, outA12〚i〛 = outAa〚i〛, outA22〚i〛 = outAa〚i〛], {i, m}]
 outA2 = DeleteCases[outA22, {0, 0, 0, 0}];
 outA3 = outAa;
 Do[A2tn = outA2〚i〛〚3〛;  (*Trial numbers from outA2*)
  hA〚A2tn〛 = 1, {i, Length[outA2]}]

In[144]:= Do[b = RandomInteger[{-179, 180}];  (*Detector 2D vector angle 1 degree increments*)
  bb = N[Flatten[{FromPolarCoordinates[{1, b * π / 180}], 0}]];
  Db = bb.Qcoordinates;  (*Convert to quaternion coordinates*)
  qb = Ls2〚i〛 ** Db;
  If[Abs[Re[qb]] > λ1〚i〛, Bb = Sign[Re[qb]], Bb = -Sign[Sin[(b - ss2〚i〛 + ξ) Degree]]];
  B0 = -Sign[Sin[(b - ss2〚i〛 + ξ) Degree]];
  outBb〚i〛 = {b, Bb, i, B0};
  If[Abs[Re[qb]] > λ1〚i〛, outB12〚i〛 = outBb〚i〛, outB22〚i〛 = outBb〚i〛], {i, m}]
 outB2 = DeleteCases[outB22, {0, 0, 0, 0}];
 outB3 = outBb;
 Do[B2tn = outB2〚i〛〚3〛;  (*Trial numbers from outB2*)
  hB〚B2tn〛 = 1, {i, Length[outB2]}]
```

Spinorial Sign Changes in A and B

For the spinorial sign changes we will need eq. (12).

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

In[148]:=
```
sscA = ConstantArray[0, m];
sscb = ConstantArray[0, m];
Do[If[hB⟦i⟧ == 1 && outAa⟦i⟧⟦2⟧ ≠ outAa⟦i⟧⟦4⟧, outA3⟦i⟧⟦2⟧ = outAa⟦i⟧⟦2⟧ * -1], {i, m}]
(*Spinorial sign change*)
Do[If[hB⟦i⟧ == 1 && outAa⟦i⟧⟦2⟧ ≠ outAa⟦i⟧⟦4⟧, ssca⟦i⟧ = 1, ssca⟦i⟧ = 0], {i, m}]
A = outA3⟦All, 2⟧;
a1 = outA3⟦All, 1⟧;
Do[If[hA⟦i⟧ == 1 && outBb⟦i⟧⟦2⟧ ≠ outBb⟦i⟧⟦4⟧, outB3⟦i⟧⟦2⟧ = outBb⟦i⟧⟦2⟧ * -1], {i, m}]
(*Spinorial sign change*)
Do[If[hA⟦i⟧ == 1 && outBb⟦i⟧⟦2⟧ ≠ outBb⟦i⟧⟦4⟧, sscb⟦i⟧ = 1, sscb⟦i⟧ = 0], {i, m}]
B = outB3⟦All, 2⟧;
b1 = outB3⟦All, 1⟧;
N[Total[ssca] / m] * 100
N[Total[sscb] / m] * 100
```

Out[158]= 3.51585

Out[159]= 3.5064

Statistical Analysis of the Particle Data Received from Alice and Bob

In[160]:=
```
theta = ConstantArray[0, m];
Do[θ = a1⟦i⟧ - b1⟦i⟧ + 361;  (*All angles are shifted by 361 degrees since θ is an index*)
 theta⟦i⟧ = θ;
 aliceD = A⟦i⟧; bobD = B⟦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, m}]
```

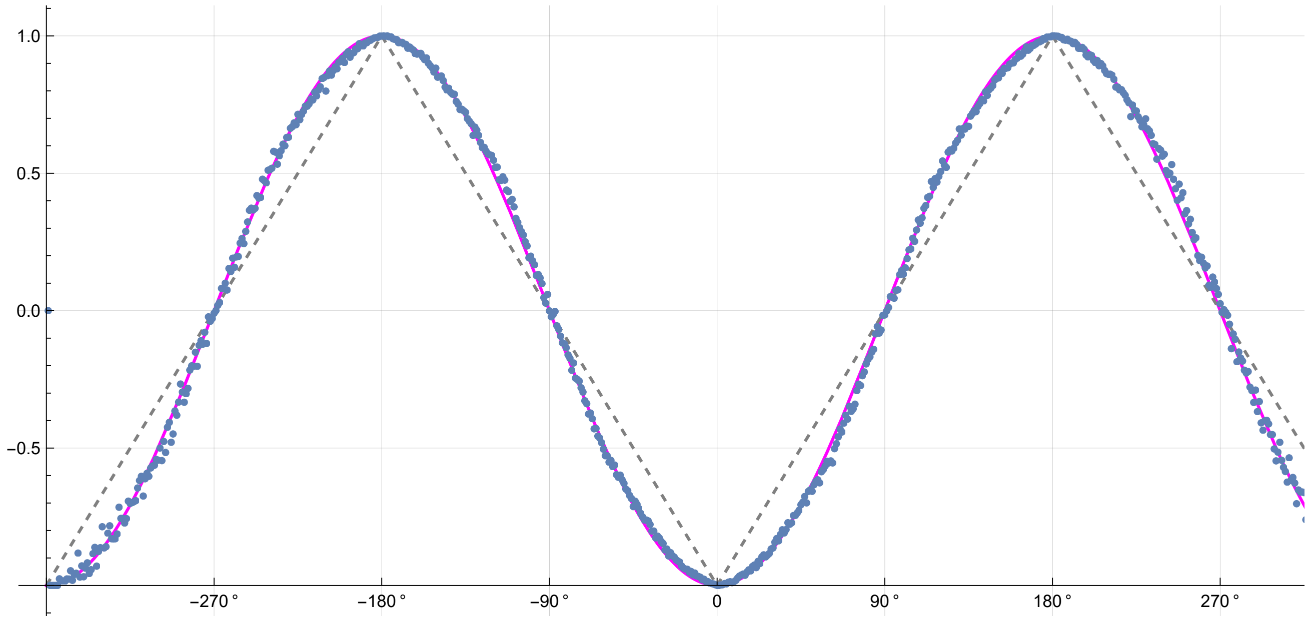
Calculating Mean Values of AB

In[162]:=
```
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}]
```

Plotting the Results Comparing Mean Values with -Cosine Curve

In[166]:=
```
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, AxesOrigin → {0, -1.0}];
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}];
Show[negcos, p1, p2, p3, p4, simulation]
```

Out[172]=



Computing Averages

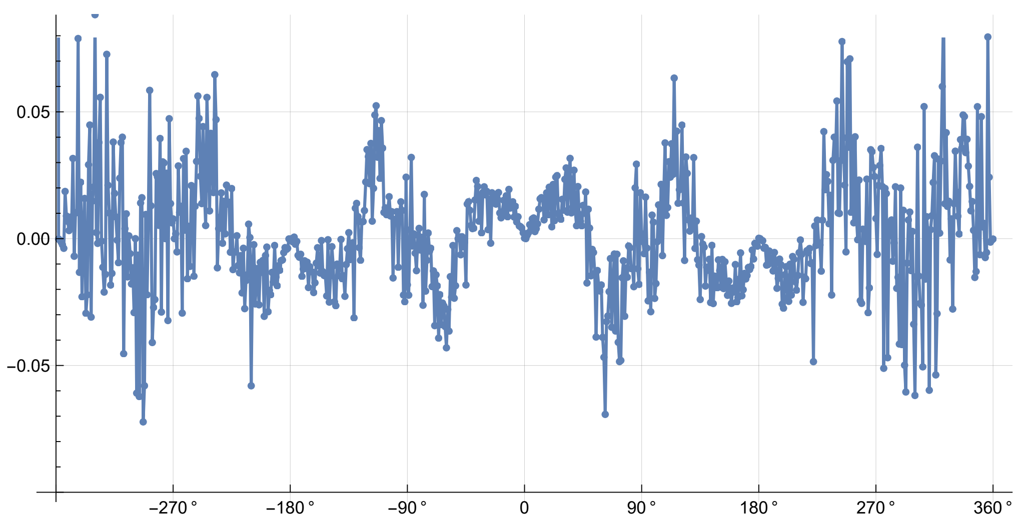
In[173]:=
```
AveA = N[Sum[A[[i]], {i, m}] / m];
AveB = N[Sum[B[[i]], {i, m}] / m];
Print["AveA = ", AveA]
Print["AveB = ", AveB]
PAP = N[Sum[nAP[[i]], {i, trialDeg}]];
PBP = N[Sum[nBP[[i]], {i, trialDeg}]];
PAN = N[Sum[nAN[[i]], {i, trialDeg}]];
PBN = N[Sum[nBN[[i]], {i, trialDeg}]];
PA1 = PAP / (PAP + PAN);
PB1 = PBP / (PBP + PBN);
Print["P(A+) = ", PA1]
Print["P(B+) = ", PB1]
totAB = Total[nPP + nNN + nPN + nNP];
Print["Total Events Detected = ", 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]
Total[PP + NN + PN + NP]
CHSH = Abs[N[mean[[315]]] - N[mean[[225]]] + N[mean[[405]]] + N[mean[[45]]]];
Print["Approx. CHSH = ", CHSH]
```

```
AveA = -0.00088

AveB = -0.000753

P(A+) = 0.49956

P(B+) = 0.499624

Total Events Detected = 2000000

Out[187]= 0.249676

Out[188]= 0.250492

Out[189]= 0.249885

Out[190]= 0.249948

Out[191]= 1.

Approx. CHSH = 2.84234
```

Deviation from negative cosine curve

```
In[211]:= 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 → 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[217]=




```
In[201]:= Mean[N[dev3]]
Mean[N[Abs[dev3]]]

Out[201]= 0.00292671

Out[202]= 0.0189798
```