Simulation Based on Michel Fodje's epr-simple and
Joy Christian's Updated 3-Sphere Model.
Parts of Quaternion and Matching Code by John Reed.
Modified by Fred Diether, Jan. 2022.

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

In[128]:=
```
<< 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 = 5 000 000;
trialDeg = 721;
ss = ConstantArray[0, m];
ss1 = ConstantArray[0, m];
λ = ConstantArray[0, m];
Ls1 = ConstantArray[0, m];
Ls2 = ConstantArray[0, m];
Da1 = ConstantArray[0, m];
Db1 = ConstantArray[0, m];
qa1 = ConstantArray[0, m];
qb1 = 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];
outB12 = Table[{0, 0, 0, 0}, m];
outA22 = Table[{0, 0, 0, 0}, m];
outB22 = Table[{0, 0, 0, 0}, m];
a1 = ConstantArray[0, m];
b1 = ConstantArray[0, m];
aa1 = ConstantArray[0, m];
bb1 = 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;
```

Generating Particle Data with Three Independent Do-Loops

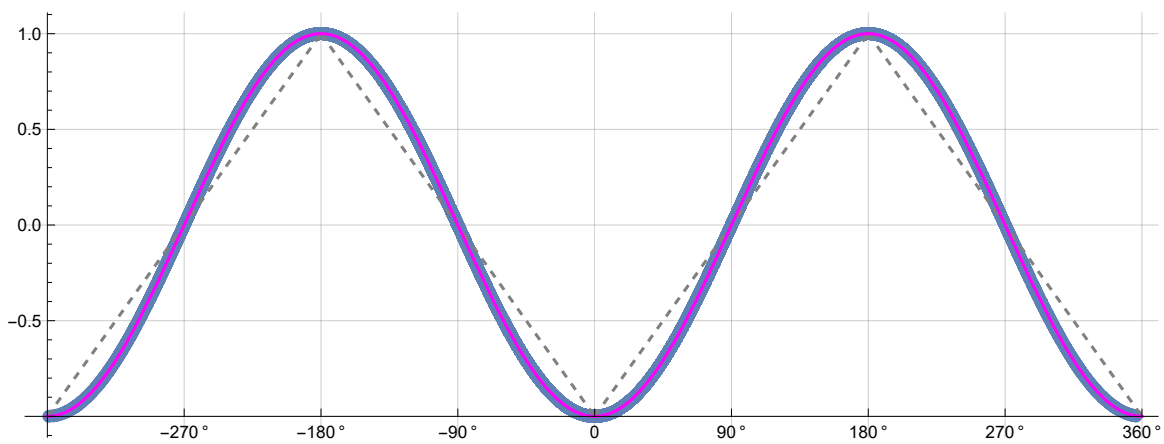
```
In[167]:= Do[s = RandomPoint[Sphere[]]; (*Singlet 3D vector*) (*Hidden Variable*)
  ss[[i]] = s;
  θ1 = ToSphericalCoordinates[s][[3]] * 180 / π;
  ss1[[i]] = θ1;
  λ[[i]] = β (Cos[θ1/ϕ]^2); (*Hidden variable mechanism*)
  Ls1[[i]] = s.Qcoordinates; (*Convert to quaternion coordinates*) (*A particle*)
  Ls2[[i]] = -s.Qcoordinates, {i, m}]   (*B particle*)

In[168]:= Do[a = RandomInteger[{-179, 180}]; (*Detector 2D vector angle 1 degree increments*)
  aa = N[Flatten[{FromPolarCoordinates[{1, a * π / 180}], 0.000001}]];
  aa1[[i]] = aa;
  Da = aa.Qcoordinates; (*Convert to quaternion coordinates*)
  a1[[i]] = a;
  qa = Da ** Ls1[[i]];
  qa1[[i]] = qa;
  If[Abs[Re[qa]] > λ[[i]], Aa = Sign[Re[qa]], Aa = Sign[Sin[(a - ss1[[i]] + ξ) Degree]]];
  A0 = Sign[N[Sin[(a - ss1[[i]] + ξ) Degree]]];
  outAa[[i]] = {a, Aa, i, A0};
  If[Abs[Re[qa]] > λ[[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[172]:= Do[b = RandomInteger[{-179, 180}]; (*Detector 2D vector angle 1 degree increments*)
  bb = N[Flatten[{FromPolarCoordinates[{1, b * π / 180}], 0.000001}]];
  bb1[[i]] = bb;
  Db = bb.Qcoordinates; (*Convert to quaternion coordinates*)
  b1[[i]] = b;
  qb = Ls2[[i]] ** Db;
  qb1[[i]] = qb;
  If[Abs[Re[qb]] > λ[[i]], Bb = Sign[Re[qb]], Bb = -Sign[Sin[(b - ss1[[i]] + ξ) Degree]]];
  B0 = -Sign[N[Sin[(b - ss1[[i]] + ξ) Degree]]];
  outBb[[i]] = {b, Bb, i, B0};
  If[Abs[Re[qb]] > λ[[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]}]
```

Verification of the Analytical 3-Sphere Model Product Calculation Based on Geometric Algebra Using Quaternions

In[176]:=
```
m2 = 20000;
r0 = ConstantArray[0, m2];
r1 = ConstantArray[0, m2];
r2 = ConstantArray[0, m2];
QAB = ConstantArray[0, m2];
plotq = Table[{0, 0}, m2];
Do[(*qA=Re[Da**Limit[Ls1⟦i⟧,Ls1⟦i⟧→Sign[Re[Da**Ls1⟦i⟧]]Da]];
 qB=Re[Db**Limit[Ls2⟦i⟧,Ls2⟦i⟧→Sign[Re[Db**Ls2⟦i⟧]]Db]];*)
 (*The above two lines are moved to the independent A and B Do-loops
   and modified for proper calculation of A and B outcomes.*)
 r1⟦i⟧ = {qa1⟦i⟧⟦2⟧, qa1⟦i⟧⟦3⟧, qa1⟦i⟧⟦4⟧};
 r2⟦i⟧ = {qb1⟦i⟧⟦2⟧, qb1⟦i⟧⟦3⟧, qb1⟦i⟧⟦4⟧};
 QAB⟦i⟧ = Re[qa1⟦i⟧] * Re[qb1⟦i⟧] - r1⟦i⟧.r2⟦i⟧;
r0⟦i⟧ = (Re[qa1⟦i⟧] Limit[Cross[s2, bb1⟦i⟧], s2 → Sign[Re[qb1⟦i⟧]] bb1⟦i⟧] +
     Re[qb1⟦i⟧] Limit[Cross[aa1⟦i⟧, s1], s1 → Sign[Re[qa1⟦i⟧]] aa1⟦i⟧] -
     Cross[Limit[Cross[aa1⟦i⟧, s1], s1 → Sign[Re[qa1⟦i⟧]] aa1⟦i⟧],
      Limit[Cross[s2, bb1⟦i⟧], s2 → Sign[Re[qb1⟦i⟧]] bb1⟦i⟧]]) /
   (Sin[ArcCos[a1⟦i⟧.b1⟦i⟧]]);
 q = {Re[QAB⟦i⟧], r0⟦i⟧⟦1⟧, r0⟦i⟧⟦2⟧, r0⟦i⟧⟦3⟧}.Qcoordinates2;
 θ = a1⟦i⟧ - b1⟦i⟧ + 360;
 plotq⟦i⟧ = {θ, Re[q]}, {i, m2}]
meanq = Mean[plotq⟦All, 2⟦]];
Print["Imaginary part vanishes. meanq = ", meanq]
sim = ListPlot[plotq, PlotMarkers → {Automatic, Small}, AspectRatio → 3 / 8,
   Ticks → {{{450, 90 °}, {90, -270 °}, {540, 180 °}, {180, -180 °}, {630, 270 °}, {270, -90 °},
      {720, 360 °}, {360, 0 °}}, 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}];
negcos = Plot[-Cos[x Degree], {x, 0, 720}, PlotStyle → {Magenta}];
Show[sim, p1, p2, p3, p4, negcos]
```

```
Imaginary part vanishes. meanq = -0.002762
```

Out[191]=



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

Spinorial Sign Changes in A and B

For the spinorial sign changes 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[192]:=
```
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⟧;
a2 = 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⟧;
b2 = outB3⟦All, 1⟧;
N[Total[ssca] / m] * 100
N[Total[sscb] / m] * 100
```

Out[202]= 3.52508

Out[203]= 3.50208

Statistical Analysis of the Particle Data Received from Alice and Bob

In[204]:=
```
theta = ConstantArray[0, m];
Do[θ2 = a1⟦i⟧ - b1⟦i⟧ + 360;
 theta⟦i⟧ = θ2;
 aliceD = A⟦i⟧; bobD = B⟦i⟧;
 If[aliceD == 1, nAP⟦θ2⟧ ++];
 If[bobD == 1, nBP⟦θ2⟧ ++];
 If[aliceD == -1, nAN⟦θ2⟧ ++];
 If[bobD == -1, nBN⟦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⟧ ++], {i, m}]
```

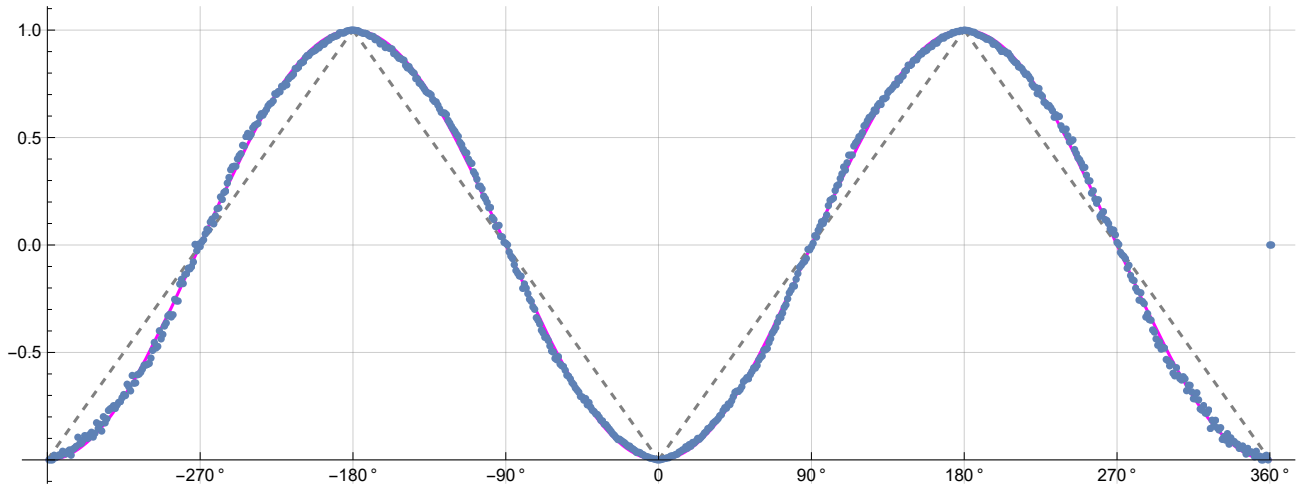
Calculating Mean Values of AB

In[206]:=
```
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[210]:=
```
sim2 = ListPlot[mean, PlotMarkers → {Automatic, Tiny}];
negcos = Plot[-Cos[x Degree], {x, 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}];
Show[negcos, p1, p2, p3, p4, sim2]
```

Out[212]=



Computing Averages

```
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 = 0.000282
AveB = 0.0004992
P(A+) = 0.500141
P(B+) = 0.50025
Total Events Detected = 5000000
```

Out[227]= 0.250153

Out[228]= 0.249762

Out[229]= 0.249988

Out[230]= 0.250097

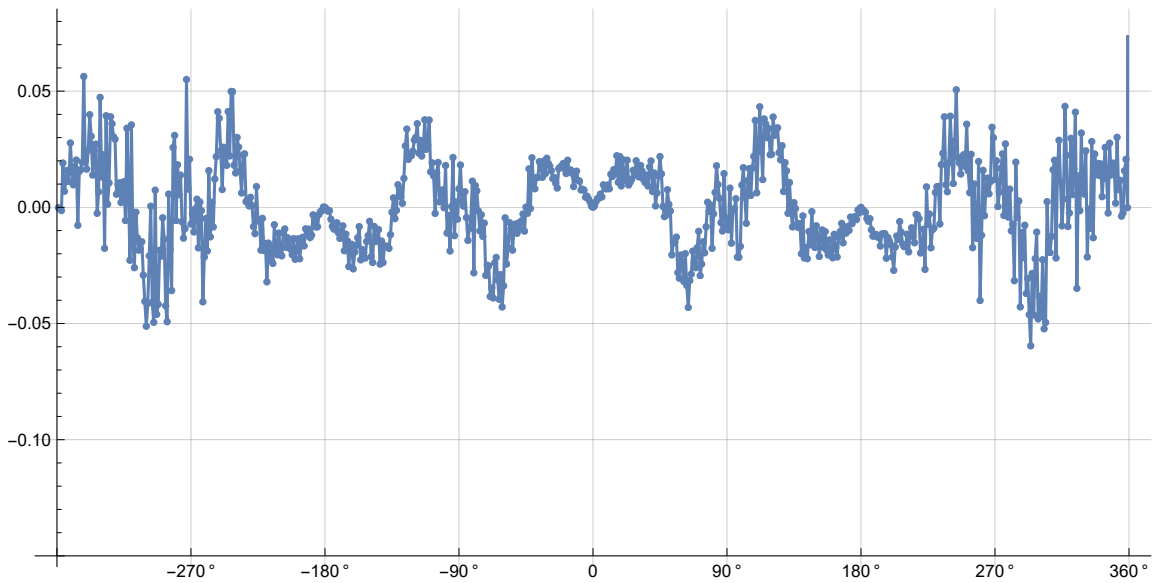
Out[231]= 1.

Approx. CHSH = 2.79335

Deviation from negative cosine curve

In[251]:=
```
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]];
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, -0.15}]
```

Out[257]=



In[241]:=
```
Mean[N[dev3]]
Mean[N[Abs[dev3]]]
```

Out[241]= 0.0014594

Out[242]= 0.0171438

```
-0.00521477; 0.0227552; (*1 million deviations*)
```