Simulation of Joy Christian's original 3-Sphere model modified to work with no orientation λ for m >>1.

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

```
<< 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 = 500 000;  (*Number of events to perform*)
s1 = ConstantArray[0, m];
ss = ConstantArray[0, m];
λ = ConstantArray[0, m];
Ls1 = ConstantArray[0, m];
Ls2 = ConstantArray[0, m];
a4 = ConstantArray[0, m];
b4 = ConstantArray[0, m];
Daa = ConstantArray[0, m];
Dbb = ConstantArray[0, m];
aa1 = ConstantArray[0, m];
bb1 = ConstantArray[0, m];
qa1 = ConstantArray[0, m];
qb1 = ConstantArray[0, m];
qA = ConstantArray[0, m];
qB = ConstantArray[0, m];
λa = ConstantArray[0, m];
λb = ConstantArray[0, m];
λ1 = ConstantArray[0, m];
λ2 = ConstantArray[0, m];
m2 = 20 000; (*Events to perform for product calculation*)
pcab = ConstantArray[0, m2];
plotpc = 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[122]:=
```
Do[s = RandomPoint[Sphere[]];  (*Singlet 3D vector; Hidden Variable*)
 s1[[k]] = s;
 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[123]:=
```
Do[a = RandomInteger[{-179, 180}];  (*Detector 2D vector angle 1 degree increments*)
 a4[[k]] = a;
 aa = N[Flatten[{FromPolarCoordinates[{1, a * π / 180}], 0.000001}]];
 aa1[[k]] = aa;
 Da = aa.Qcoordinates;  (*Convert to quaternion; A detector*)
 Daa[[k]] = Da;
 qa = Da ** Ls1[[k]];  (*Detector - particle interaction*)
 qa1[[k]] = qa;
 If[Abs[Re[qa]] > λ[[k]], qA[[k]] = Sign[Re[qa]],
  qA[[k]] = Sign[Sin[(a - ss[[k]] + ξ) Degree]]];
 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*)
```

In[124]:=
```
Do[b = RandomInteger[{-179, 180}];  (*Detector 2D vector angle 1 degree increments*)
 b4[[k]] = b;
 bb = N[Flatten[{FromPolarCoordinates[{1, b * π / 180}], 0.000001}]];
 bb1[[k]] = bb;
 Db = bb.Qcoordinates;  (*Convert to quaternion; B detector*)
 Dbb[[k]] = Db;
 qb = Ls2[[k]] ** Db;  (*Detector - particle interaction*)
 qb1[[k]] = qb;
 If[Abs[Re[qb]] > λ[[k]], qB[[k]] = Sign[Re[qb]],
  qB[[k]] = -Sign[Sin[(b - ss[[k]] + ξ) Degree]]];
 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*)
```

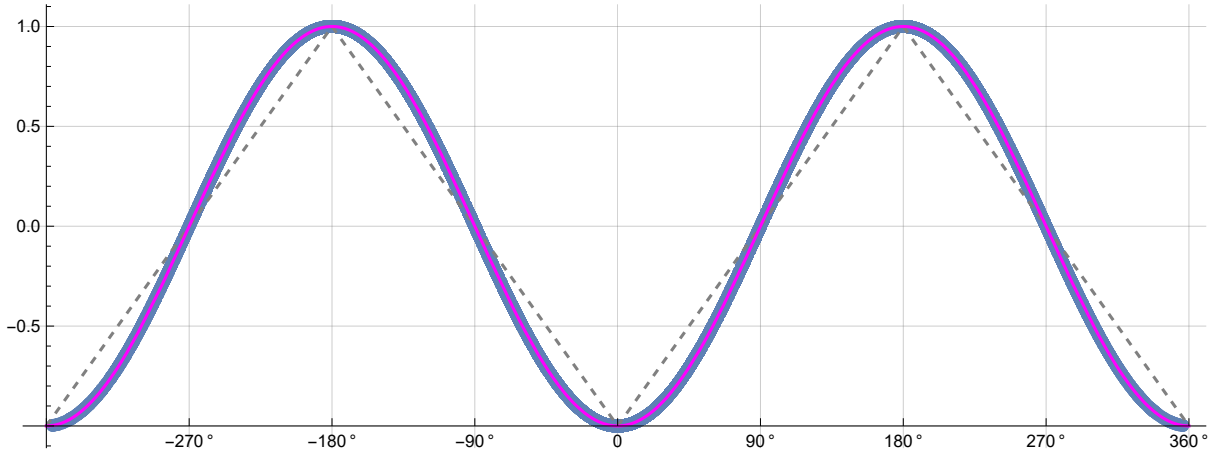
Verification of the 3-Sphere Model Product Calculation Prediction

In[125]:=
```
(*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[pc = Limit[Daa[[k]] ** Ls1[[k]] ** Ls2[[k]] ** Dbb[[k]],
    {Ls1[[k]] → Sign[aa1[[k]].s1[[k]]] Daa[[k]], Ls2[[k]] → Sign[bb1[[k]].s1[[k]]] Dbb[[k]]}];
 pcab[[k]] = FromQuaternion[pc];
 Θ = a4[[k]] - b4[[k]];
 plotpc[[k]] = {Θ, Re[pc]}, {k, m2}]
```

In[126]:=
```
meanpc = N[Expand[Mean[pcab]]];
Print["Imaginary part vanishes. meanpc = ", meanpc];
sim1 = ListPlot[plotpc, 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]
```

Imaginary part vanishes. meanpc = $\left(0.00436878 + 8.25607\times10^{-9}\ \mathrm{i}\right) - 1.0655\times10^{-9}\ \mathrm{J} - 0.000464092\ \mathrm{K}$

Out[134]=



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

In[135]:=
```
Atot = ConstantArray[0, m];
Btot = ConstantArray[0, m];
Do[If[λ2[[k]] == k && qA[[k]] ≠ λa[[k]], A = -qA[[k]], A = qA[[k]]];
 (*Spinorial sign change correction*)
 Atot[[k]] = A;
 If[λ1[[k]] == k && qB[[k]] ≠ λb[[k]], B = -qB[[k]], B = qB[[k]]];
 (*Spinorial sign change correction*)
 Btot[[k]] = B;
 If[λ2[[k]] == k && qA[[k]] ≠ λa[[k]], ssca[[k]] = 1, ssca[[k]] = 0];
 If[λ1[[k]] == k && qB[[k]] ≠ λb[[k]], sscb[[k]] = 1, sscb[[k]] = 0];
θ2 = a4[[k]] - b4[[k]] + 360;  (*Angles shifted by 360 since θ2 is an index*)
 If[A == 1, nAP[[θ2]]++];
 If[B == 1, nBP[[θ2]]++];
 If[A == 1 && B == 1, nPP[[θ2]]++];
 If[A == 1 && B == -1, nPN[[θ2]]++];
 If[A == -1 && B == 1, nNP[[θ2]]++];
 If[A == -1 && B == -1, nNN[[θ2]]++], {k, m}]
PercentForm [N[Total[ssca] / m]] (*Percentage of spinorial sign changes.*)
PercentForm[N[Total[sscb] / m]]
```

Out[138]//PercentForm=
3.413%

Out[139]//PercentForm=
3.377%

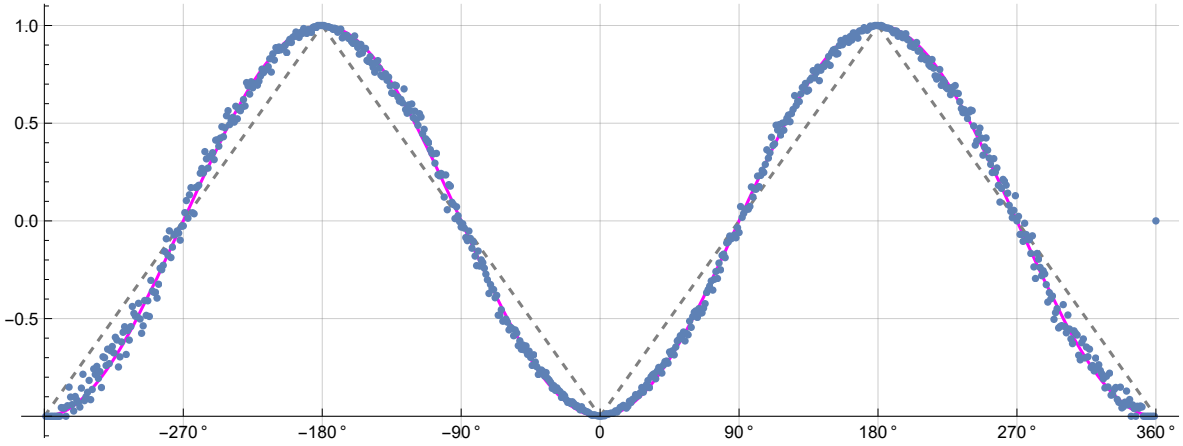
Calculating Mean Values of AB

In[140]:=
```
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[144]:=
```
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[148]=



Computing Total Events Detected and Averages

In[149]:=
```
totAB = Total[nPP + nNN + nPN + nNP];
AveA = N[Total[Atot] / totAB];
AveB = N[Total[Btot] / 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 = 500 000

AveA = −0.0006

AveB = −0.002616

P(A+) = 0.4997

P(B+) = 0.498692

P(++) = 0.249076

P(−−) = 0.250684

P(+−) = 0.250624

P(−+) = 0.249616

Approx. CHSH = 2.7004