## Simulation Based on Michel Fodje's epr-simple simulation translated from Python to Mathematica by John Reed 13 Nov 2013
## Modified by Fred Diether for Completely Local-Realistic Dec. 2021
## Using John Reed's trial number matching code.

### Set Run Time Parameters, Initialize Arrays and Tables

In[150]:=
```
m = 10 000 000;
trialDeg = 721;
s = ConstantArray[0, m];
λ = ConstantArray[0, m];
outA = Table[{0, 0, 0, 0, 0}, m];
outB = Table[{0, 0, 0, 0, 0}, m];
outAa = Table[{0, 0, 0, 0, 0}, m];
outBb = Table[{0, 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];
η = 3;
β = 0.3;
ξ = -15;
```

### Generate Particle Data with 3 Do Loops

In[171]:=
```
Do[e = RandomReal[{0, 360}];  (*Singlet vector angle*)
 s[[i]] = e;
 λ[[i]] = β (Cos[e/η]^2),  (*Hidden Variables*)
 {i, m}]
```

In[172]:=
```
Do[a = RandomInteger[{0, 360}];  (*Detector vector angle 1 degree increments*)
 If[Abs[Cos[((a - s[[i]]) Degree)]] < λ[[i]], C1 = f1, C1 = g1];
 If[Abs[Cos[((a - s[[i]]) Degree)]] > λ[[i]],
  Aa = -Sign[Cos[((a - s[[i]]) Degree)]], Aa = -Sign[Sin[((a - s[[i]] + ξ) Degree)]]];
 A0 = -Sign[Sin[((a - s[[i]] + ξ) Degree)]];
 outAa[[i]] = {a, Aa, i, C1, A0}, {i, m}]
outA1 = Select[outAa, MemberQ[#, g1] &];
outA2 = Select[outAa, MemberQ[#, f1] &];
```

In[175]:=
```
Do[b = RandomInteger[{0, 360}];  (*Detector vector angle 1 degree increments*)
 If[Abs[Cos[((b - s⟦i⟧) Degree)]] < λ⟦i⟧, C2 = f2, C2 = g2];
 If[Abs[Cos[((b - s⟦i⟧) Degree)]] > λ⟦i⟧,
  Bb = Sign[Cos[((b - s⟦i⟧) Degree)]], Bb = Sign[Sin[((b - s⟦i⟧ + ξ) Degree)]]];
 B0 = Sign[Sin[((b - s⟦i⟧ + ξ) Degree)]];
 outBb⟦i⟧ = {b, Bb, i, C2, B0}, {i, m}]
outB1 = Select[outBb, MemberQ[#, g2] &];
outB2 = Select[outBb, MemberQ[#, f2] &];
```

## Match Events Using Trial Numbers and do Spinorial Sign Changes

In[178]:=
```
outA3 = outAa;
outB3 = outBb;
tnA = ConstantArray[0, m];
tnB = ConstantArray[0, m];
ssca = ConstantArray[0, m];
sscb = ConstantArray[0, m];
Do[tnB2 = outB2⟦i⟧⟦3⟧; (*Trial numbers from B2*)
 tnB⟦tnB2⟧ = 1, {i, Length[outB2]}]
Do[If[tnB⟦i⟧ == 1 && outAa⟦i⟧⟦2⟧ ≠ outAa⟦i⟧⟦5⟧, outA3⟦i⟧⟦2⟧ = outAa⟦i⟧⟦2⟧ * -1], {i, m}]
(*Spinorial sign change.*)
Do[tnA2 = outA2⟦i⟧⟦3⟧; (*Trial numbers from A2*)
 tnA⟦tnA2⟧ = 1, {i, Length[outA2]}]
Do[If[tnA⟦i⟧ == 1 && outBb⟦i⟧⟦2⟧ ≠ outBb⟦i⟧⟦5⟧, outB3⟦i⟧⟦2⟧ = outBb⟦i⟧⟦2⟧ * -1], {i, m}]
(*Spinorial sign change.*)
Do[If[tnB⟦i⟧ == 1 && outAa⟦i⟧⟦2⟧ ≠ outAa⟦i⟧⟦5⟧, ssca⟦i⟧ = 1, ssca⟦i⟧ = 0], {i, m}]
Do[If[tnA⟦i⟧ == 1 && outBb⟦i⟧⟦2⟧ ≠ outBb⟦i⟧⟦5⟧, sscb⟦i⟧ = 1, sscb⟦i⟧ = 0], {i, m}]
A = outA3⟦All, 2⟧;
B = outB3⟦All, 2⟧;
a1 = outA3⟦All, 1⟧;
b1 = outB3⟦All, 1⟧;
N[Total[ssca] / m] * 100 (*Percent of total spinorial sign changes*)
N[Total[sscb] / m] * 100
```

Out[194]= 4.94891

Out[195]= 4.94482

## Statistical Analysis of Particle Data

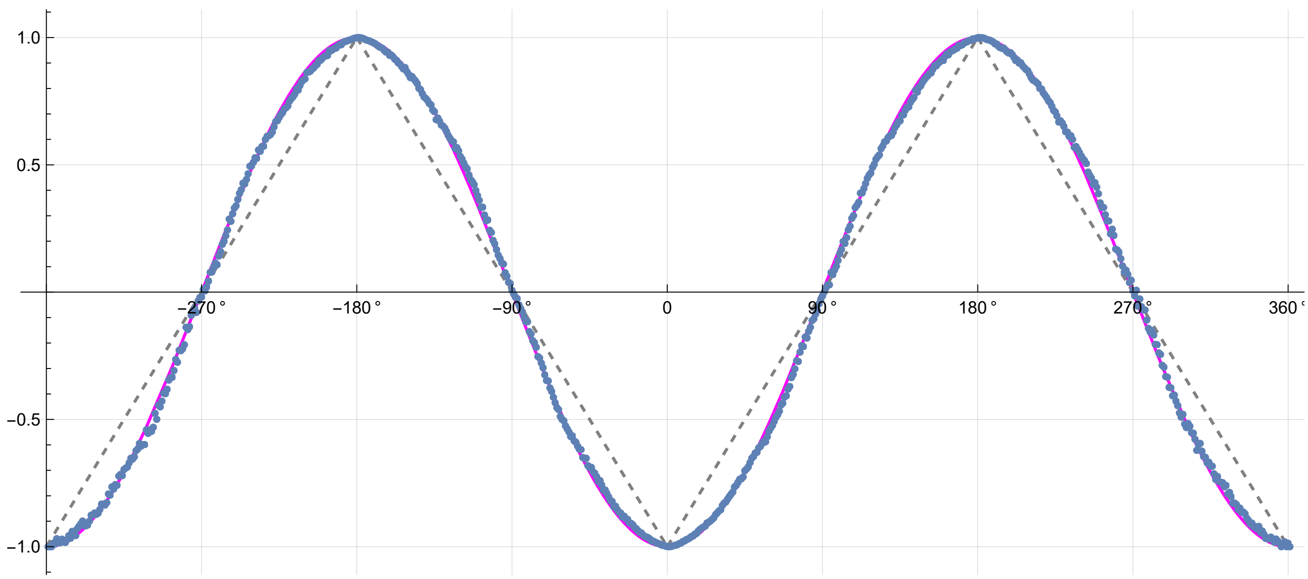
In[196]:=
```
Do[θ = a1⟦j⟧ - b1⟦j⟧ + 361;  (*All angles are shifted by 2π since θ is an index*)
 aliceD = A⟦j⟧; bobD = B⟦j⟧;
 If[aliceD == 1, nAP⟦θ⟧++];
 If[bobD == 1, nBP⟦θ⟧++];
 If[aliceD == -1, nAN⟦θ⟧++];
 If[bobD == -1, nBN⟦θ⟧++];
 If[aliceD == 1 && bobD == 1, nPP⟦θ⟧++];
 If[aliceD == 1 && bobD == -1, nPN⟦θ⟧++];
 If[aliceD == -1 && bobD == 1, nNP⟦θ⟧++];
 If[aliceD == -1 && bobD == -1, nNN⟦θ⟧++], {j, m}]
```

## Calculate Mean Values and Plot

In[197]:=
```
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[201]:=
```
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}];
Show[negcos, p1, p2, p3, p4, simulation]
```

Out[207]=




## Calculate Averages, Approximate CHSH

In[208]:=
```
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 = Sum[nPP[[i]] + nNN[[i]] + nPN[[i]] + nNP[[i]], {i, trialDeg}]
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]
CHSH = Abs[N[mean[[315]]] - N[mean[[225]]] + N[mean[[405]]] + N[mean[[45]]]]
```

AveA = 0.0005374

AveB = 0.0005382

P(A+) = 0.500269

P(B+) = 0.500269

Out[220]= 10 000 000

Out[221]= 0.250193

Out[222]= 0.249656

Out[223]= 0.250075

Out[224]= 0.250076

Out[225]= 2.79607