Event-driven simulation of Frustrated Interference with a local model.

Pierre Leroy

The paper presents a local event-driven simulation method of an interference experiment that generates a violation of Bell's inequality while demonstrating the absence of quantum entanglement. It describes a mechanism based on the assumption of non-interacting states of light (or 'dark states') and the modulation of these states by interference when photon wave packets overlap. The simulation shows that this mechanism can produce non-classical visibility, cosine coincidences, and a CHSH violation consistent with experimental results.

Introduction.

A recent experiment by Wang et al. (August 2025) [1] demonstrated a significant violation of the CHSH inequality, while explicitly showing that this phenomenon could not be attributed to quantum entanglement.

This result challenges the prevailing view that Bell inequality violations are exclusively a consequence of non-local quantum entanglement (often termed 'spooky action at a distance'). The experiment suggests that the violation stems from quantum indistinguishability, prompting a re-examination of the role of local realism in such quantum correlation experiments.

In this paper, we propose an alternative, Local Hidden-Variable (LHV) explanation. We hypothesize that the common cause is related to the existence of non-interacting states of light [4]. These states, which cannot be registered by detectors, create a systematic and phase-dependent detection loophole, resulting in a non-equitable sampling of the measured correlations.

The paper presents a local method using these states to reproduce the experimental results.

1. Description of the experiment. (brief summary)

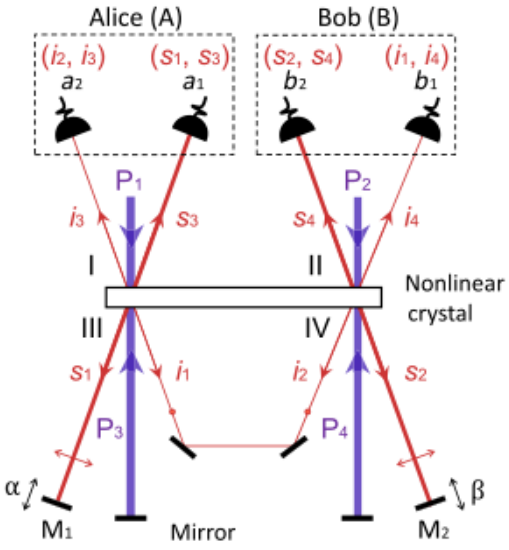


Fig 1. (taken from the original document [1] Fig.1.B)

The experiment uses an interferometer that generates so-called 'frustrated' interference.
Alice and Bob's arms each contain two detectors.
An event is detected when both detectors of an arm each detect a photon, i.e. a pair measurement on a1a2 or b1b2.
A coincidence is counted if Alice and Bob each detect a pair, i.e., a triggering of all four detectors.

Two movable mirrors M1 and M2 allow the optical path length of photons s1 and s2 to be modified, thus generating an adjustable phase shift for these photons.

2. Interpretation through quantum mechanics.

This describes the observed CHSH violation (S=2.342±0.051) as the consequence of an indistinguishability of paths and sources that produced the quadruple detection.
This indistinguishability is then the condition which allows the sources to generate the pairs necessary for the coincidence measurement.
SPDC sources can then only generate pairs for quadruple a1a2b1b2 detection if the mirror positions allow for indistinguishability.
In other words, the sources must 'know' at SPDC time whether the mirror positions allow them to generate a photon pair.

3. Locally realistic interpretation used for the simulation.

In the local simulation presented, the pairs are emitted independently of the position of the mirrors.

The basic assumptions used to produce the simulation are as follows:

- The existence of dark and light states of light (PDS/MSS states [4]).
  In a dark (PDS) state, the photon cannot interact with the detector.
  The PDS/MSS state can reverse during interference between two photons. This depends on its phase, its polarization, and a local LHV variable.
- Interference between two photons occurs when the wave packets of the two photons physically overlap.
  For this overlap to be possible, the optical path difference between the two photons must be less than the coherence length of the photons. In this context, path indistinguishability is not the cause of interference, but a prerequisite for interference to occur.
  A second necessary assumed condition is that the two photons cross inside the parametric conversion crystal and pre-exist before the crossing. Interference does not occur if one of the two photons is SPDC generated during the passage through the crystal.
  In the experiment, this condition is only met by the photon pairs i1+s2 and i2+s1.

<u>One additional hypothesis</u>.

The essential assumption, allowing the experiment to be simulated locally, in addition to that of the existence of dark states, is the following:

- Every pair emitted by SPDC contains two photons in inverted PDS/MSS states.

This means that one of the two photons in the pair will not be detectable and that a single SPDC conversion will only be able to produce a single measurement.

This implies that, if no interference is produced, the only way to produce a quadruple coincidence measurement requires that each of the four sources produces a pair so that four photons are in a detectable state and that these four photons are each directed to a different detector.
This natively produces a very low probability of quadruple coincidence in g^4.

If interference between photons occurs, the probability of obtaining two detectable states increase and depends on the alpha and beta phase settings.

4. Simulation process.

The core of the process is the strict adherence to local realism and causality. All photon properties are defined at the source (SPDC), and any subsequent interaction (interferometry) only depends on the local properties of the involved photons. There is no information transfer between Alice's and Bob's stations.

The photon model uses 3 LHVs:

- variable e (boolean): describes the PDS/MSS state. (0/1)
- variable p: H or V polarization.
- variable q: a stochastic variable [-1..1] (a physical interpretation is omitted in this document).

4.1 Simulation steps:

➢1. On a pump pulse, the 4 sources have a probability g of generating a pair.
If a pair is generated by a source, the LHVs of the pair are initialized :
The variables e for signal/idler are defined with opposite states. (one bright, one dark)
The variables p are defined at H for signal, V for idler.
The variables q are defined with a common random value.
Also, a random emission time is defined, as it is assumed that the four sources do not generate a pair at exactly the same instant. This time introduces phase jitter during the interference between photons.

➢2. For photons s1 and s2, reflected by mirrors M1 and M2, the optical path length is adjusted according to the alpha and beta settings respectively. This alters the phase difference when s1 or s2 interferes with another photon.

➢3. Simulation of interference.
If signal and idler photons pass together through the SPDC crystal and their wave packets overlap, the detectability state of the two photons is altered as a function of their phase difference, polarization and the local variable q.

➢4. Detection and counting:
Only detectable MSS photons are counted. (50% of those emitted)

4.2 Simulation difficulties:

There are two difficulties:

➢1. Produce a non-classical visibility > $1/\sqrt{2}$ of quadruple coincidences a1a2b1b2, while maintaining a constant detection rate of a1a2 and b1b2 pairs.

➢2. Produce non-classical (non-triangular) quadruple coincidences, but in the form of a cos(alpha+beta) according to equation 2 of the document: k*[2 + 2*cos(alpha+beta)]

Obtaining CHSH > 2 occurs if both of these points are met.

Point 1 seems impossible to solve if we postulate that the pairs a1a2 and b1b2 are produced independently and by chance, because to increase the probability of measuring a quadruple coincidence, we must necessarily increase the number of pairs a1a2 and b1b2 produced.

Point 2, producing cosine correlations and dependence on the sum of angles alpha+beta, which appears non-local, statistically emerges from the effect of unfair sampling and the effect of LHV q during interference.

5. Simulation results.

2*10^6 pump pulses are simulated.
The code simulates ideal detectors, detecting 100% of detectable photons (MSS) .

5.1 Pair rate and visibility.

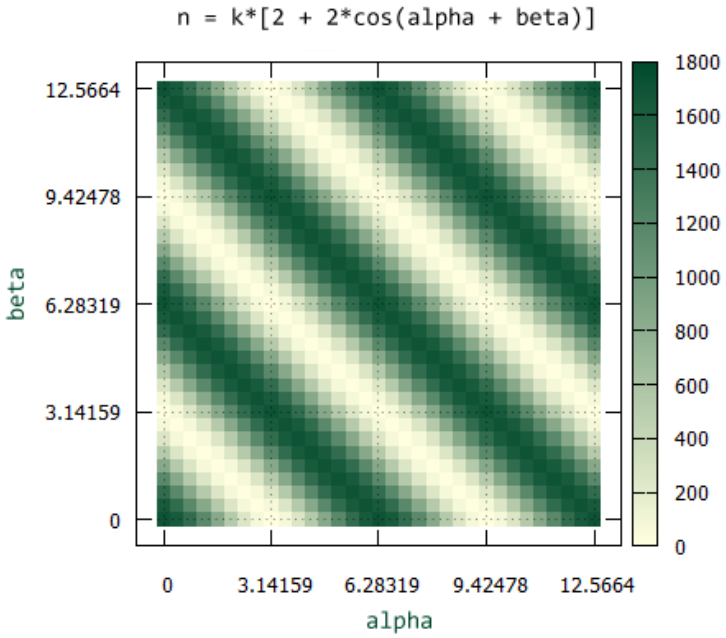
```
Test1: check pair rate constant [0..2*pi]..
alpha:0.00  beta:0.00  182174 184027 182660 182322   5731 5736   3382
alpha:0.00  beta:0.79  183666 182938 183252 182826   5875 5852   2940
alpha:0.00  beta:1.57  183452 182434 182929 183385   5747 5790   1748
alpha:0.00  beta:2.36  182720 183022 182710 183007   5817 5719   769
alpha:0.00  beta:3.14  182805 182550 183226 183402   5832 5859   376
alpha:0.00  beta:3.93  183071 183004 183128 183134   5709 5769   741
alpha:0.00  beta:4.71  183484 182878 182972 183033   5755 5746   1773
alpha:0.00  beta:5.50  183341 182968 183167 183356   5764 5714   2895
alpha:0.00  beta:6.28  183417 183149 183425 183013   5731 5765   3374
min_cc:376.0  max_cc:3382.0  vis: 0.800
alpha:0.79  beta:0.00  182867 183916 183026 182964   5694 5791   2878
alpha:0.79  beta:0.79  183127 182597 183427 182345   5635 5653   1657
alpha:0.79  beta:1.57  183153 184097 184165 182566   5740 5827   736
alpha:0.79  beta:2.36  183381 183424 183029 182552   5858 5727   435
alpha:0.79  beta:3.14  183150 183229 182676 183418   5860 5722   748
alpha:0.79  beta:3.93  183253 182527 182751 182820   5802 5833   1754
...
...
alpha:6.28  beta:0.00  183566 182374 182364 183405   5715 5788   3524
alpha:6.28  beta:0.79  183021 183170 183227 183461   5812 5796   2926
alpha:6.28  beta:1.57  183410 183023 182711 182825   5641 5674   1712
alpha:6.28  beta:2.36  183187 183147 182607 183056   5879 5679   773
alpha:6.28  beta:3.14  183392 183201 182601 182983   5669 5731   401
alpha:6.28  beta:3.93  182818 182998 182725 183052   5786 5852   764
alpha:6.28  beta:4.71  182890 183095 182850 182617   5778 5702   1732
alpha:6.28  beta:5.50  183192 183393 182968 182390   5847 5769   2894
alpha:6.28  beta:6.28  182683 182841 183253 183412   5916 5843   3572
min_cc:401.0  max_cc:3572.0  vis: 0.798
Average visibility: 0.797
```

The number of quadruple coincidence varies between approximately 350 and 3500 depending on alpha+beta settings.

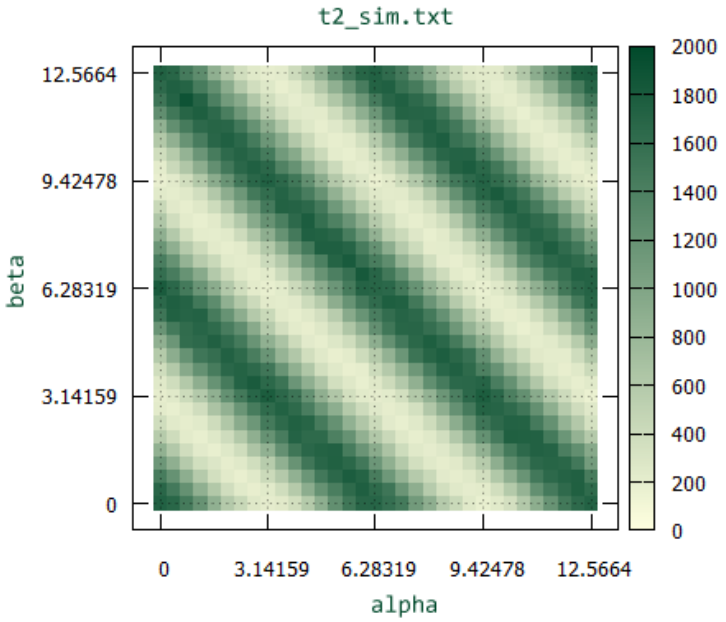
Visibility is greater than $1/\sqrt{2} \approx 0.707$ and fluctuates around 0.8
The count of single measurements and pairs is globally constant (see note 1).

5. 2 Form of coincidences.

The quadruple coincidence rate must vary in k*[2+2*cos(alpha+beta)], in accordance with equation eq(2) of the experiment document [1]: $N(\alpha, \beta) = g^4 * N0 * [2+2\cos(\alpha+\beta)]$.
A visual comparison is made by plotting this theoretical function with the one obtained by simulation.



Graph 2.1: Theoretical function: 2+2cos(α+β) [0..4π]



Graph 2.2: simulated result. [0..4π]

We can see that the gradients of the theoretical graph and those of the simulation are very close. Only the contrast changes, because the visibility of the simulation (approximately 0.8) is lower.

5.3 Evaluation of CHSH.

The calculation method defined in the experiment document is used.
The results obtained, generated with the theoretical equation eq(2) and those simulated, are as follows:

```
t3: eval CHSH..
CHSH (QM theoretical) = 2.8309
CHSH (simulated) = 2.3499   (experiment expect S=2.342 +/- 0.051)
```

We note that the simulation generates a result within the margin of error calculated experimentally in the document 'Supplementary Materials, Fig S4' [3], despite the simulation of perfect detectors.
This may indicate that the maximum value predicted by quantum mechanics of $2\sqrt{2}$ might be impossible to reach with this experiment if we assume the existence of undetectable states.

5.4 Hidden non-locality check.

To detect a possible non-locality erroneously introduced into the simulation code, a test was performed by disabling conditional detection dependent on the photon state e.
All emitted photons are then detected, and unfair sampling is disabled.
We then observe that the cos(alpha+beta) interference disappears completely, visibility becomes zero, and the CHSH value approaches 0. This confirms that the results obtained are solely a consequence of unfair sampling.

Note 1.
The simulation is entirely local, however, a common state of the local variable e is defined at emission for the four sources, defined randomly at each pump pulse.
We can assume, to remain local, that this common initialization is dependent on the properties of the pump beam.
With this random initialization, the rate of simple measurements is identical for all detectors.

It is possible to eliminate this dependence on the pump beam by defining a constant value of e for the signal and idler photons (but opposite).
However, this has the effect of generating an imbalance in the number of simple measurements measured on each detector, without however producing an effect on the number of pairs measured which remains constant.

We then measure muchs less simple measurements on detectors a2 and b1 ('external' Fig.1) than on detectors a1 and b2 ('internal').

```
Test1: check pair rate constant [0..2*pi]..
alpha:0.00  beta:0.00  359131 6123 6151 359621   5744 5759   3429
alpha:0.00  beta:0.79  360432 6216 6039 360361   5813 5672   2890
alpha:0.00  beta:1.57  360252 6125 6175 360083   5763 5842   1721
alpha:0.00  beta:2.36  360125 6251 5978 359965   5886 5602   741
alpha:0.00  beta:3.14  359555 6048 6109 359820   5688 5777   405
alpha:0.00  beta:3.93  359372 6212 6225 360089   5873 5876   802
alpha:0.00  beta:4.71  360625 6160 6203 359769   5830 5876   1813
alpha:0.00  beta:5.50  359546 6185 6214 359750   5817 5866   2977
alpha:0.00  beta:6.28  360636 6188 6244 360833   5840 5872   3562
min_cc:405.0  max_cc:3562.0  vis: 0.796
alpha:0.79  beta:0.00  359952 5992 6038 359764   5656 5687   2860
alpha:0.79  beta:0.79  361412 6186 6110 359403   5845 5759   1754
...
```

However, using the experimental raw data [2], we also observe a significant asymmetry on the simple measurement rates of the four detectors.

```
19664903  25778121  14811933  34118297  1464816  1236557   246
19670568  25749342  14810523  34099539  1461485  1236425   409
19672450  25705549  14798541  34076463  1460386  1234531   387
19664559  25709157  14766350  34037324  1463689  1230357   296
```

Raw data from the file 'VBI_Coincidence_20230707.dat' (first 4 lines)

The differences are much less pronounced, however, as the experiment has a much lower detection efficiency than the simulation, a large number of simple measurements could attenuate the visibility of a theoretical effect.

In the simulation code included in the appendix, this constant initialization option can be enabled by setting SIM_CSIE 1 (line 20).

Conclusion.

The simulation shows that, by considering the possibility of the existence of dark states of light and some local assumptions, a structurally simple deterministic process can simulate the experimental results.
It allows for a visibility greater than $1/\sqrt{2}$ and the dependence of seemingly non-local measurements on $\cos(\alpha + \beta)$. This dependence in $\cos(\alpha + \beta)$, although evaluated nowhere in the simulation code, emerges naturally from the local filtering process.

The dark states of light, if real, could be the common cause explaining the violation of CHSH inequality by quantum indistinguishability and that produced by entangled states in EPR experiments, this by using an unavoidable detection loophole generating a specific unfair sampling.

References.

[1] « Violation of Bell inequality with unentangled photons »
https://arxiv.org/abs/2507.07756
[2] « Github resources »
https://github.com/NJU-Malab/Violate-CHSH-in-FI
[3] « Supplementary Materials for Violation of Bell inequality with unentangled photons »
sciadv.adr1794_sm.pdf
[4] « "Bright and dark states of light: The quantum origin of classical interference" »
https://arxiv.org/abs/2112.05512

Appendice.
Simulation source (C code) (direct download here : if_chsh.c)

```
// ---------------------------------------
// Local simulation of experiment "Violation of Bell inequality with unentangled photons"
// https://arxiv.org/abs/2507.07756
// Simulation assume dark states of light, decribed in paper:
// "Bright and dark states of light: The quantum origin of classical interference"
// https://arxiv.org/abs/2112.05512
// GCC build:
// gcc -O2 if_chsh.c -o if_chsh.exe
// author: pierrel5556@gmail.com date: 9 nov 2025
// ---------------------------------------
#include <stdio.h>
#include <math.h>

typedef int bool;        // 0/1 boolean type

#define PI 3.14159265358979323846

#define SPDC_PROB 0.096  // paper experiment SPDC efficiency
#define E_DET 1          // detectable bright state
#define SIM_CSIE 0       // if 1: simulate constant signal/idler e value (see paper)

// ---------------------------------------------
// RNG

// RNG seed
static unsigned long r_seed = -123;
#define UPD_SEED (r_seed = (r_seed * 214013L + 2531011L))

// return random value in [0..1[ range
static _inline float rand1u(void)
{
  return (1.0f/0x10000)*(UPD_SEED >> 16);
}

// return signed random value in [-1..1[ range
static _inline float rand1s(void)
{
  return (1.0f/0x10000)*((int)UPD_SEED >> 15);
}

// return random boolean state 0/1
static _inline bool rand_bool(void)
{
  return (int)UPD_SEED >= 0;
}

// ----------------------------
// counters type

struct ctr_t
{
  // Alice counters
  int a1;              // single
```

```
  int a2;              // single
  int a1a2;            // pair

  // Bob counters
  int b1;
  int b2;
  int b1b2;

  // quads
  int a1a2b1b2;
};

// ----------------------------
// photons LHV

struct pho_t
{
  int e;                // detectability state 0/1
  double p;
  double q;

  // simulation variables (not LHV)
  bool gen;             // flag generated by spdc
  double dl;            // delta path length (emit time jitter)
};

// ----------------------------
// laser pulse simulation

#define H_POL 0
#define V_POL (PI/2)

// SPDC init
static _inline void sim_SPDC(struct pho_t *s, struct pho_t *i, bool es)
{
  if (rand1u() < SPDC_PROB)  // SPDC efficiency
  {
    // init signal
    s->e = es;
    s->p = 0;          // H polarization
    s->q = rand1s();   // q: random

    // init idler
    i->e = !es;        // always inverted e state
    i->p = PI/2;       // V polarization
    i->q = s->q;       // same as signal

    // simulation variables (not LHV)
    s->gen = 1;        // flag is generated
    i->gen = 1;        // flag is generated
    s->dl = rand1s();  // delta path = time emit jitter * c
    i->dl = s->dl;     // delta path, same as signal
  }
  else
  {
    // no SPDC occured
    s->gen = 0;
```

```
    i->gen = 0;
  }
}

// Simulate interference effect on photon state e
static _inline void photon_IF(struct pho_t *pho, double phi)
{
  double d = 2.0*(phi - pho->p);
  double w = cos(d + asin(pho->q));
  if (w < -0.5)
    pho->e = !pho->e;
}

// Simulate interference between 2 photons ()
// Is local only if photons wave packet can overlap (path indistinguishability
consequence)
static _inline void pair_IF(struct pho_t *p1, struct pho_t *p2)
{
  if (p1->gen && p2->gen)          // both SPDC generated
  {
    double phi = (p1->dl - p2->dl)*PI;    // phase difference
    photon_IF(p1, phi);
    photon_IF(p2, phi);
  }
}

// Simulate single pump pulse
static void sim_pulse(double alpha, double beta, struct ctr_t *ctr)
{
  struct pho_t s1, i1, s2, i2, s3, i3, s4, i4;  // photons LHV

  // emit
#if SIM_CSIE == 1
  bool r0 = 1;                    // simulate constant signal/idler e
#else
  bool r0 = rand_bool();          // use signal/idler random values (pump dependant)
#endif

  // simulate the 4 sources
  sim_SPDC(&s1, &i1, r0);
  sim_SPDC(&s2, &i2, r0);
  sim_SPDC(&s3, &i3, r0);
  sim_SPDC(&s4, &i4, r0);

  // add alpha beta delta path length (*2)
  s1.dl += alpha*2;
  s2.dl += beta*2;

  // Simulate interference between signal idler.
  // Assumed to occur only if signal/idler share same path through the SPDC cristal.
  // Do not occur if one photon of the 2 photon is generated at interference position by
SPDC
  // Only i2s1 and i1s2 meet these conditions.
  pair_IF(&i2, &s1);
  pair_IF(&i1, &s2);

  // detection and count
```

```
  {
    // macro: detector click if photon was SPDC generated and in bright state
    #define DET(p1, p2) ((p1.gen && (p1.e == E_DET)) || (p2.gen && (p2.e == E_DET)))

    // detect bright states
    int det_a1 = DET(s1, s3);
    int det_a2 = DET(i2, i3);
    int det_b1 = DET(i1, i4);
    int det_b2 = DET(s2, s4);

    // update counters
    if (det_a1) ctr->a1++;
    if (det_a2) ctr->a2++;
    if (det_b1) ctr->b1++;
    if (det_b2) ctr->b2++;

    if (det_a1 && det_a2) ctr->a1a2++;
    if (det_b1 && det_b2) ctr->b1b2++;
    if (det_a1 && det_a2 && det_b1 && det_b2) ctr->a1a2b1b2++;
  }
}

// simulate N pump pulses using alpha and beta phases
static void sim_stage(double alpha_rad, double beta_rad, int N, struct ctr_t *ctr)
{
  // convert phase angles to mirrors moves
  double alpha = alpha_rad/(4*PI);
  double beta = beta_rad/(4*PI);
  int i;
  for (i=0; i<N; i++)
    sim_pulse(alpha, beta, ctr);
}

// return quad coincidences count for N pump pulses
static int get_cc4(double alpha_rad, double beta_rad, int N)
{
  if (N > 0)             // simulate if N > 0
  {
    struct ctr_t ctr = { 0 };
    sim_stage(alpha_rad, beta_rad, N, &ctr);
    return ctr.a1a2b1b2;
  }
  else                   // return theoretical value
  {
    double k = -N*5*(SPDC_PROB*SPDC_PROB*SPDC_PROB*SPDC_PROB);  // 5: arbitrary value
    return (int)(k*(2+2*cos(alpha_rad+beta_rad)));  // paper eq(2)
  }
}

// ----------------------------------------
// tests and graphes
// ----------------------------------------

// ----------------
// step 1:
// - check pairs count is constant and independant of quad coincidences
// - check visibility is > sqrt(2)/2
```

```
static void test1(int N)
{
  double alpha;
  double an_max = 2*PI + 1e-10;
  double vis_sum = 0;
  double vis_cnt = 0;
  for (alpha=0; alpha <= an_max; alpha += PI/4)
  {
    double cc_min = N;
    double cc_max = 0;
    double beta, vis;
    for (beta=0; beta <= an_max; beta += PI/4)
    {
      struct ctr_t ctr = { 0 };
      sim_stage(alpha, beta, N, &ctr);
      if (ctr.a1a2b1b2 < cc_min) cc_min = ctr.a1a2b1b2;
      if (ctr.a1a2b1b2 > cc_max) cc_max = ctr.a1a2b1b2;
      printf("alpha:%.2f  beta:%.2f  %d %d %d %d   %d %d   %d\n", alpha, beta, ctr.a1,
ctr.a2, ctr.b1, ctr.b2, ctr.a1a2, ctr.b1b2, ctr.a1a2b1b2);
    }
    vis = (cc_max - cc_min)/(cc_max + cc_min);
    vis_sum += vis;
    vis_cnt += 1;
    printf("min_cc:%.1f  max_cc:%.1f  vis: %.3f\n", cc_min, cc_max, vis);
  }
  printf("Average visibility: %.3f\n", vis_sum/vis_cnt);  // seem to be 0.8
}

// ----------------
// step 2: check, using graph that:
// - quads coincidences follow k*[2 + 2*cos(alpha+beta)]
// - compare visually with theoretical graph

static void test2(const char *file_name, int N)
{
  FILE *f = fopen(file_name, "wb");
  if (f)
  {
    double alpha;
    double an_max = 4*PI + 1e-10;
    printf("generate gnuplot graph file: %s\n", file_name);
    for (alpha=0; alpha <= an_max; alpha += PI/8)
    {
      double beta;
      printf("alpha: %.2f\n", alpha);    // progression
      for (beta=0; beta <= an_max; beta += PI/8)
      {
        int  n_cc = get_cc4(alpha, beta, N);
        fprintf(f, "%.2f %.2f %d\n", alpha, beta, n_cc);
      }
    }
    fclose(f);
  }
}

// ----------------
// step 3:
```

```
// Check CHSH > 2.0

static double get_E(double alpha, double beta, int N)
{
  double N_pp = get_cc4(alpha     , beta     , N); // N(+1, +1)
  double N_pm = get_cc4(alpha     , beta + PI, N); // N(+1, -1)
  double N_mp = get_cc4(alpha + PI, beta     , N); // N(-1, +1)
  double N_mm = get_cc4(alpha + PI, beta + PI, N); // N(-1, -1)
  double D = N_pp + N_pm + N_mp + N_mm;
  // E = P(+1,+1) - P(-1,+1) - P(+1,-1) + P(-1,-1)
  if (D > 0.0)
    return (N_pp - N_mp - N_pm + N_mm) / D;
  return 0;
}

static void get_CHSH(int N)
{
  const double alpha1 = 0.0;
  const double alpha2 = PI / 2.0;
  const double beta1  = PI / 4.0;
  const double beta2  = 3.0 * PI / 4.0;

  double E11 = get_E(alpha1, beta1, N);
  double E12 = get_E(alpha1, beta2, N);
  double E21 = get_E(alpha2, beta1, N);
  double E22 = get_E(alpha2, beta2, N);

  // S = |-E(a1, b1) + E(a1, b2) + E(a2, b1) + E(a2, b2)| (Eq. 8) [9]
  double S_raw = -E11 + E12 + E21 + E22;
  double S = fabs(S_raw);
  if (N <= 0)
    printf("CHSH (QM theoretical) = %.4f\n", S);
  else
    printf("CHSH (simulated) = %.4f   (experiment expect S=2.342 +/- 0.051)\n", S);
}


// ------------------
// main.
// if N set <= 0, theoretical result is used (eq(2))
// if N set > 0, simulation is used using N pulses.

int main(void)
{
#if 1
  printf("------------------------------\n");
  printf("Test1: check pair rate constant [0..2*pi]..\n");
  test1(2000000);
#endif

#if 1
  printf("------------------------------\n");
  printf("Test2: generate CC4 [0..4*pi]..\n");

  test2("t2_eq2.txt", -1000000);      // splot "t2_eq2.txt" with image
  test2("t2_sim.txt",  1000000);      // splot "t2_sim.txt" with image
```

```
  // gnuplot commands:
  // cd 'C:\dev_c\phy\sim_FI'
  // set pm3d map;set size ratio -1;set palette defined (0 "#ffffdf", 1 "#00492b")
  // set xtics pi;set ytics pi;set xrange[-pi/4:4*pi+pi/4];set yrange[-pi/4:4*pi+pi/4]
  // splot "t2_eq2.txt" with image
  // splot "t2_sim.txt" with image
#endif

#if 1
  printf("------------------------------\n");
  printf("Test3: eval CHSH..\n");
  get_CHSH(-1000000);
  get_CHSH(1000000);
  return 0;
#endif
}
```

Version 1.0, nov 10 2025.