A detectability principle for event detection in EPR experiment simulations.

Pierre Leroy

Abstract:

This document describes a non-local method to produce detection events during an EPR experiment simulation using photons.
It is based on a local realistic model using a principle of detectability, on which acts a non-local influence.
A computer simulation completes the description.

Introduction:

Quantum mechanics does not describe a mechanism for producing events.
The method described here use a causal process that is deterministic.
It makes it possible to show that at least one process is possible to simulate an experiment, and can help to produce hypotheses on the functioning of the real event production process.

1/ Description of the process.

It uses 3 main parts:

- A principle of detectability.
- A non-local influence.
- A local polarizer simulation model.

Description of the parts:

- Principle of detectability:

  It assumes that the particles have a state that will define whether they are detectable by a detector.
  This state is modified by some specific physical interactions.
  If the particle is not detectable, it will not be able to produce detection, even if the detection efficiency is 100%.
  This non-detectability is of theoretical origin.

- Non-local influence:

  This influence invert the state of detectability of one of the particles of a pair.
  The choice of the particle is randomly chosen.

- Polarizer:

  The simulation of the polarizer is done using a local model based on LHV. [1]
  This uses basically 2 LHV, to which is added for the model described here an additional LHV defining the state of detectability.

- Detectors:

  Ideal detectors are simulated, but they can only detect a particle if its state is detectable.

2/ General operation.

This part describes the general functioning of a simulation, without going into details.

It is noted 's' the state of detectability of a particle.

It is contained in an LHV and can take two values, with:

s = 1 : detectable particle.
s = 0 : non-detectable particle.

Source:

The source emits a pair of particles, and defines for each photon of the pair a random state of detectability s=0 or s=1 with an average proportion of 50% for each state.

To define a free state, the state s of the two particles must be defined by independent random states s.
(s Alice = r1, s Bob = r2, r1/r2 independent random values)

To define an entangled state, the state s of the two particles must be defined by an identical random state for the two particles. (s Alice = s Bob = r)

Polarizer:

It works locally and orients the particle towards two detectors depending on the polarization of the particle and the angle of the polarizer.

Under certain circumstances, it inverts the s-state of the particle.

Non-local effect.

In certain circumstances, depending on a type of interaction which has occurred for the two particles at the level of their polarizer, it inverts the state of one of the particles of the pair.
The choice of the particle is random.

Detection.

If the state of the particle corresponds to the state detectable by the detector, the particle can be detected. Otherwise, the detector will not trigger, regardless of its sensitivity.

The choice of the detectable state can be fixed defined at 0 or 1.
Both value select a group of particles that produce the same correlations.
In the following it will assumed arbitrarily that detectors can be triggered only if the particle state s is 1.

Summary:

We have a system that can detect certain particles depending on a state.

This state is defined at emission, and can be reversed by a local interaction with a polarizer, and a non-local influence.

3/ Operation details.

This part describes the operation from the LHV of the particles.

Each particle has 3 LHVs.

- p: polarization angle. [-PI..PI]
- q: unidentified quantity [-1..1]
- s: 0 or 1 state of detectability.

During the interaction of the particle with its polarizer, its polarization is modified.
This polarization variation value is denoted **rp** by the model.

If the amplitude of this variation exceeds PI/4, the state of detectability s is reversed.

if |rp| > PI/4 then s = not s (state reversal)

The non-local effect is triggered in a particular condition depending on the two interactions that have taken place at the level of the two polarizers.

This condition corresponds to situations where the two values of rp of repolarization of the particles are negative.

By noting **tg** the triggering of the non-local effect : [0 or 1]

tg = (rp(Alice)< 0) and (rp(Bob) < 0)

If tg is activated, one of the two particles is chosen at random, and its state is reversed.

At Detection, the particle is detectable if its state s is the same as that detectable by the detector.

Then if particle LHV s is 1 (1 : detectable state by detector), an detection event is produced.

If at least one of the particles cannot be detected, the emitted pair cannot produce a coincidence measurement, and will produce either a single measurement or no measurement (uu).

4/ Correlations and CHSH.

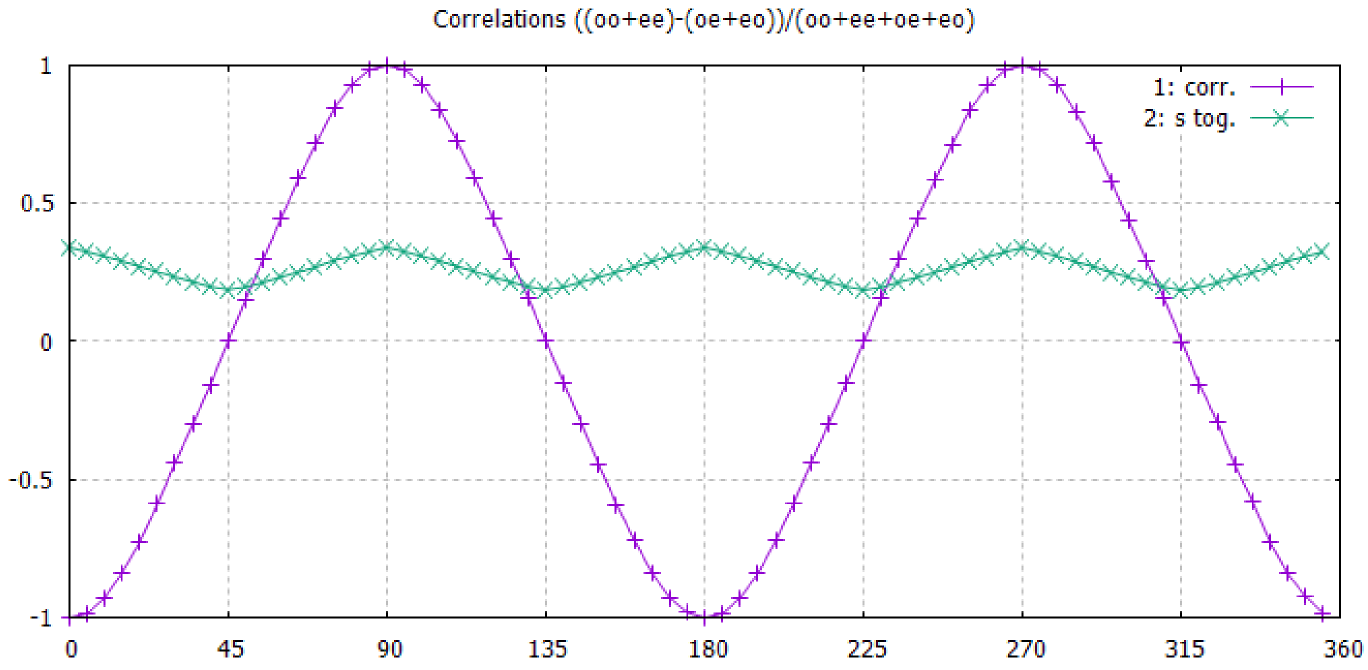
The following two graphs display the correlations produced by the whole simulation mechanism.

<u>Graph 1</u>:

It displays the correlations produced by defining entangled particles.

This is done by defining the 3 values of LHV at emission with the following conditions :

- p Bob = p Alice. (or p Alice + constant offset, PI/2 is used for graph)
- q Bob = q Alice. (or q Alice + constant offset)
- s Bob = s Alice. (same detectability state)



The x axis displays the angle difference between the polarizers.

The y-axis displays for Curve 1 the correlations produced by the detectable pair measurements using the following correlation formula:

E = (N_eq – N_nq)/(N_eq + N_nq).

With:
N_eq: number of measurements having produced identical events for the two photons (++ --, or oo ee)
N_nq: number of measurements having produced different events for the two photons (+- -+, or oe eo)

Since only detectable pairs are counted, E evaluates apparent correlations.
The value of E would be different if all the particles were detectable, and would then evaluate the real correlations produced by all the emitted pairs.

The curve 2 displays the rate of photons [0..1] whose initial detectability state defined at emission has been reversed either by the interaction with the polarizer, or by the non-local influence.

It can be noticed that the total amount of photons changing state of detectability between emission and detection is on average 25% and has a triangular appearance.

CHSH:

The CHSH can take 2 possible values depending on whether the emission polarization is defined fixed, or random.
A random polarization is simulated by adding an identical random angle to the LHV p values of Alice and Bob.
Both options produce the same correlation graph but this affects the CHSH value .

```
Fixed  polarization: CHSH = 2.999972 (3 probable)
Random polarization: CHSH = 2.575084 (+/- 0.00015 estimated)
```

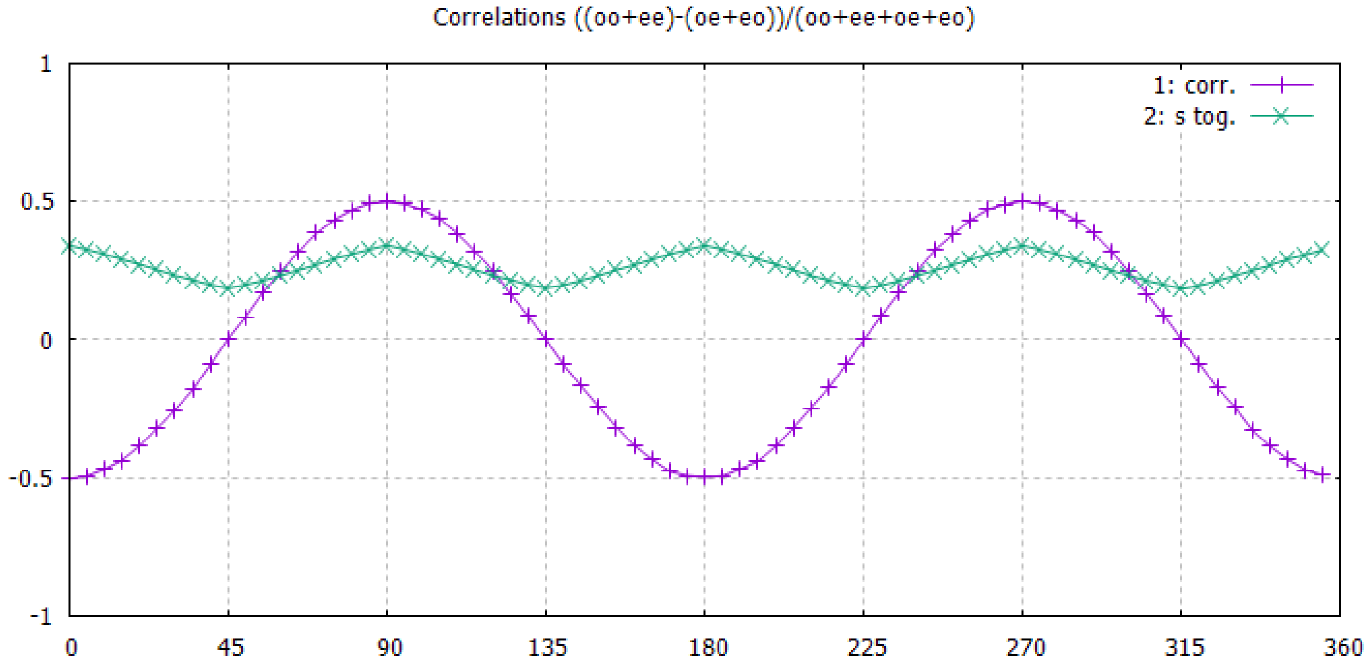
note: N used for CHSH calculation is 500,000,000

Graph 2:

It displays the correlations produced by defining free particles.

This is done by initializing independent random values for the LHVs s and q of the two particles.
The p polarizations LHVs values are initialized the same way as to simulate entangled particles.

p Bob = p Alice. (or p Alice + constant offset, PI/2 is used for graph)
q Alice = random value [-1..1]
s Alice = random value 0 or 1
q Bob = random value [-1..1]
s Bob = random value 0 or 1



It can be noticed that the proportion of photons whose state of detectability has changed is the same as for the entangled particles.

CHSH:

```
Fixed polarization : CHSH = 1.414..
Random polarization: CHSH = 1.414..
```

The CHSH result is identical in both cases.

5/ Discussion.

Real experiment simulation

This detectability mechanism influenced by a non-local effect only detects a proportion of the particles emitted.

It produces an unavoidable proportion of simple measurements and measurements containing no detections (uu).

One might think in the first place that it will not be able to reproduce the results of real experiments.

However, it was initially assumed for the simulation of such an experiment [2] where too many detections were observed compared to the source emission rate configuration.

The simulation of this effect then allowed to solve this problem.

This mechanism also has another important property.

It make possible to reproduce certain graphics produced from real raw data which have a local appearance, although a non-local effect acts.

It also allows to produce a violation of the Bell inequalities which take into account the single detections with the correct violation value. (Eberhard inequality)

Is s local or non-local ?

The non-local influence only has a detectable effect on the correlations if the particles are emitted with identical s-states.

‘s’ has been defined so far as a local variable, modifiable by a non-local influence.

We could have defined it as a non-local variable, shared by the two particles of an entangled pair.

However, as the non-local mechanism acts in the same way on all the particles, it would no longer work by simulating free photons, because it is then necessary to have different s values for the two non-entangled particles in order to observe correct correlations.

Nature of non-local influence

The fact that correlations can be defined using only LHVs implies that there is no requirement that there be a link between two particles of a pair. This eliminates the need for an instantaneous or superlight-speed influence between the s-state of the particles.

Only a static configuration of an interaction produced by the space is necessary.

The influence which affects ‘**s**’ can then come from an interaction with the space in which the particle moves and that affect in the same way all the particles which cross it.

It is then necessary that the spatial configuration which defines this influence is established according to the experimental configuration, mainly the angle between the polarizers.

To explain the experiments which modify the angle configuration of the polarizers after emission of a pair, it is then necessary to suppose that the configuration of the influence is established at a speed higher than c.

Conclusion:

The principle of detectability associated with a non-local influence makes it possible to produce apparent correlations close to those predicted by QM.

This mechanism produces a strong non-fair sampling effect during measurements.

According to this principle, reality seems to be composed of two sets of particles with only one that is observable and which exchange their particles during specific interactions.

This exchange rate seems to average 25% in EPR interactions with photons.

This principle makes it possible to produce particles with a free or entangled appearance solely as a function of the initialization of the LHV values at emission of a pair.

The magnitude of the measurable correlations then only depends on the LHV values.

The non-local influence affects all particles, free and entangled, in the same way.

There is then no physical difference between a pair of free and entangled particles, because the interactions on the particles are the same in all cases.

The effect of the non-local influence could then be explained by a spatial influence established by the experimental configuration.

This simulation method, mainly algorithmic, requires a theoretical physical interpretation.
However, as it is deterministic, such an interpretation seems possible.

References:

[1]: The local polarizer model used for the simulation.
https://www.vixra.org/abs/2201.0019

[2]: Simulation of a real experiment using this model.
http://pierrel5.free.fr/physique/sim_g2013/sim_ev_real_en.html

Computer simulation (C language)

This code is used to produce graphs 1 and 2 using the gnuplot utility.
It also calculates the CHSH.

Two options make it possible to select between the emission of entangled or free photons, and an emission with a fixed or random polarization.
The gnuplot command line to display the graph is commented out at the end of the source code.

The code can also be downloaded directly here.
http://pierrel5.free.fr/physique/det_pr/det_epr_nl.c

Source code:

```
// This code show an example of using particule detectability states coupled to a non local influence to simulate EPR correlations.

#include <stdio.h>                               // for print functions
#include <math.h>                                // for math acos() function
#include <float.h>

#define PI 3.14159265358979323846
#define DEG_TO_RAD(d) (((d)*PI)/180.0)           // convert degree to radians

// ------------------------------------------------
// simulate polarizer

#define OUT_U 0                                  // coded value for undetected
#define OUT_O 1                                  // coded value for o output
#define OUT_E 2                                  // coded value for e output

// results of sim_polarizer() call
int pol_out;                                     // photon polarizer output o/e
double pol_rp;                                   // photon repolarization value [-PI/2..PI/2]

// Simulate polarizer.
// Described here: https://www.vixra.org/abs/2201.0019
// procedure arguments:
// pol_angle : angle of polarizer in radians.
// lhv_p : photon lhv polarization.
// lhv_q : photon lhv q.
// define results in pol_out and pho_st global variables.

void sim_polarizer(double pol_angle, double lhv_p, double lhv_q)
{
  // get photon polarization / polarizer angle difference modulus PI
  double ad = fmod(lhv_p - pol_angle, PI);

  // get s: [-PI..PI+PI/2] range
  double s = ad + acos(lhv_q)*0.5;

  // define pol_out and pol_rp
       if (s >=  PI)    { pol_out = OUT_O; pol_rp = ad -   PI; }
  else if (s >=  PI/2)  { pol_out = OUT_E; pol_rp = ad - PI/2; }
  else if (s >=  0)     { pol_out = OUT_O; pol_rp = ad;        }
  else if (s >= -PI/2)  { pol_out = OUT_E; pol_rp = ad + PI/2; }
  else                  { pol_out = OUT_O; pol_rp = ad +   PI; }
}
```

```
// ----------------------------------------------
// random generator for EPR test

// RNG seed
unsigned long r_seed = -123;

// return signed random value in -1..1 range
double rand1s(void)
{
  r_seed = r_seed * 214013L + 2531011L;
  return ((short)(r_seed >> 16))*(1.0/0x8000);
}

// ----------------------------------------------
// EPR simulation

// --------------------------
// some simulation options

// source options
int opt_emit_fixed_pol = 0;                      // set to 1 to simulate fixed emit
polarization
int opt_emit_free_pho = 0;                       // set to 1 to simulate free photons

// --------------------------
// counters for graph

// emitted pair counter for each angle difference between polarizer
int ctr_n_pair[360 + 1] = { 0 };                 // emitted pair count

// events counters, initialized with 0
int ctr_eq[360 + 1] = { 0 };                     // oo + ee events
int ctr_neq[360 + 1] = { 0 };                    // oe + eo events

// counter for graph
int ctr_tog_st[360 + 1] = { 0 };   // particles s state toggle between emission and detection
int ctr_sing[360 + 1] = { 0 };     // single event counter ou + uo + eu + ue
int ctr_uu[360 + 1] = { 0 };       // uu counter (no detection at Alice and Bob)

// --------------------------

// Rotate Alice and Bob polarizers between 0..2*PI
// Memorize event correlation counters to draw correlation graph.

// Count of simulated events in EPR test for each angle combination
#define N_EPR 1000                  // count of emitted pair pour each angle diff settings

#define AN_STEP_DEG 5               // step in filters angles difference to produce graph

// simulate N_EPR emit with an_pol_a/an_pol_b polarizer angles
void epr_n_test(double an_pol_a, double an_pol_b, int ad)
{
  // produce N_EPR events
  int n;
  ctr_n_pair[ad] += N_EPR;
  for (n=0; n<N_EPR; n++)
  {
    // declare lhv alice and bob
    double a_lhv_p, b_lhv_p;                     // lhv polarization
    double a_lhv_q, b_lhv_q;                     // lhv q
    int a_lhv_s, b_lhv_s;                        // lhv s detectability state
    int a_s0, b_s0;                              // lhv s initial emit state (graph usage)

    // declare polariser simulation results for alice and bob
    int a_out, b_out;                            // polarizer out
    double a_rp, b_rp;                           // repolarization

    // ----------------------------
    // simulate emission at source.
    // init lhv for Alice/Bob.

    // lhv Alice
    if (opt_emit_fixed_pol)      // simulate fixed emit polarization (polarized source)
      a_lhv_p = DEG_TO_RAD(0.0); // init Alice polarization with a fixed angle value
    else
      a_lhv_p = rand1s()*PI;     // init Alice polarization with +/- PI random polarization
    a_lhv_q = rand1s();                          // random q [-1..1]
    a_lhv_s = rand1s() < 0 ? 0 : 1;              // random s 0 or 1

    // lhv Bob
    b_lhv_p = a_lhv_p + PI/2;  // Bob polarization same as Alice + constant angle difference.
    if (opt_emit_free_pho)                       // simulate free photons
    {
      b_lhv_q = rand1s();                        // Bob q independant of Alice q value
      b_lhv_s = rand1s() < 0 ? 0 : 1;            // random s 0 or 1
    }
    else
    {
      b_lhv_q = a_lhv_q;                         // Bob q = Alice q value
      b_lhv_s = a_lhv_s;                         // Bob s = Alice s value
    }

    // save emit initial state (graph)
    a_s0 = a_lhv_s;
    b_s0 = b_lhv_s;

    // ----------------------------
    // simulate Alice

    // simulate, update pol_out and pol_rp
    sim_polarizer(an_pol_a, a_lhv_p, a_lhv_q);

    // get results
    a_out = pol_out;                             // polarizer output o or e
    a_rp = pol_rp;                               // get rp
    if (fabs(pol_rp) >= PI/4)                    // test if |rp| >= PI/4
      a_lhv_s = !a_lhv_s;                        // inverse s

    // ----------------------------
    // simulate Bob

    sim_polarizer(an_pol_b, b_lhv_p, b_lhv_q);

    // get results
    b_out = pol_out;
    b_rp = pol_rp;
    if (fabs(pol_rp) >= PI/4)
```

```
      b_lhv_s = !b_lhv_s;                        // inverse s

    // simulate non local influence
    if ((a_rp < 0) && (b_rp < 0))                // both particle rp < 0
    {
      if (rand1s() < 0)                          // reverse s randoly for Alice or Bob
        a_lhv_s = !a_lhv_s;
      else
        b_lhv_s = !b_lhv_s;
    }

    // simulate detection using detectability
    // if particle state not match detector detectable state, particule is not detected

    #define DETECT_ST 1                          // define the state that detectors can detect

    if (a_lhv_s != DETECT_ST)
      a_out = OUT_U;

    if (b_lhv_s != DETECT_ST)
      b_out = OUT_U;

    // ----------------------------
    // update counters for graph

    if ((a_out == OUT_U) && (b_out == OUT_U))    // count no detection (uu)
      ctr_uu[ad]++;                              // increment +1 uu counter
    else
    if ((a_out == OUT_U) || (b_out == OUT_U))    // count single
      ctr_sing[ad]++;                            // increment +1 single counter
    else
    if (a_out == b_out)                          // ++ or -- (oo or ee)
      ctr_eq[ad]++;             // increment +1 oo/ee coincidence counter (++ + --)
    else
      ctr_neq[ad]++;            // increment +1 oe/eo or eo/oe coincidence counter (+- + -+)

    // update state change counter
    if (a_lhv_s != a_s0)
      ctr_tog_st[ad]++;
    if (b_lhv_s != b_s0)
      ctr_tog_st[ad]++;
  }
}

// --------------------------
// EPR test
void epr_test(void)
{
  int a, ad;
  for (a=0; a<360; a++)
  {
    double an_pol_a = (a*PI)/180.0;              // polarizer angle at Alice in radians
    for (ad=0; ad<360; ad+=AN_STEP_DEG)          // angle differnce
    {
      double ad_rad = (ad*PI)/180.0;             // Bob polarizer angle
      double an_pol_b = an_pol_a + ad_rad;
      epr_n_test(an_pol_a, an_pol_b, ad);        // produce N_EPR events for ad
    }
  }
```

```
}

// --------------------------
// get CHSH
// https://en.wikipedia.org/wiki/CHSH_inequality

static double eval_E(double an_pol_a, double an_pol_b)
{
  int i, Ntot;
  double E = 0;

  // reset counters, store results in ctr[0]
  ctr_eq[0] = 0;
  ctr_neq[0] = 0;

  for (i=0; i<5000; i++)                         // repeat base EPR test to increase accuracy
    epr_n_test(an_pol_a, an_pol_b, 0);

  Ntot = (ctr_eq[0] + ctr_neq[0]);               // total count of measured pairs
  if (Ntot)                                      // ensure not 0 for division
    E = (double)(ctr_eq[0] - ctr_neq[0]) / Ntot;
  return E;
}

void chsh_test(void)
{
  double a1 = 0;
  double a2 = (45.0 * PI)/180.0;
  double b1 = (22.5 * PI)/180.0;
  double b2 = (67.5 * PI)/180.0;

  double E_a1b1 = eval_E(a1, b1);
  double E_a1b2 = eval_E(a1, b2);
  double E_a2b1 = eval_E(a2, b1);
  double E_a2b2 = eval_E(a2, b2);

  double S = E_a1b1 - E_a1b2 + E_a2b1 + E_a2b2;
  double CHSH = fabs(S);

  printf("CHSH: %f\n", CHSH);
}

// ----------------------------------------
// Produce graph data file for gnuplot

// produce datas file to draw correlations graph
static void gen_gnuplot_file_co(const char *name)
{
  int ad, detect_sum = 0;

  FILE *f = fopen(name, "wb");
  if (!f)                                        // check file created
  {
    printf("Error: cannot create file %s\n", name);
    return;
  }

  // Save datas to graph file
  for (ad=0; ad<360; ad+=AN_STEP_DEG)
  {
    // correlations
    double eq  = ctr_eq[ad];
    double neq = ctr_neq[ad];
    double co = ((eq + neq) > 0) ? (eq - neq)/(eq + neq) : 0; // ensure enough measurements

    // photon toggle ratio between emission and detection (rp + nl effects combined)
    double r_tog_st = (double)ctr_tog_st[ad]/(2*ctr_n_pair[ad]);

    // measurement single ratio [0..1] (uu ignored)
    double r_sing = ctr_sing[ad] ? (double)ctr_sing[ad]/(ctr_sing[ad] + eq + neq) : 0;

    // uu/emit pair ratio [0..1]
    double r_uu = (double)ctr_uu[ad]/ctr_n_pair[ad];

    // save results for graph
    fprintf(f, "%d %.3f %.3f %.3f %.3f\n", ad, co, r_tog_st, r_sing, r_uu);
  }
  fclose(f);
}

// main program entry
int main(void)
{
  r_seed = -123;                                 // RNG seed

  // define emission options
  opt_emit_fixed_pol = 1;   // set to 1 to simulate fixed emit polarization (crystal out)
  opt_emit_free_pho = 0;    // set to 1 to simulate free photons

  printf("Start EPR simulation, please wait..\n");
  epr_test();

  // define graph file for gnuplot
  gen_gnuplot_file_co("gr_co.txt");

  // get CHSH
  printf("Get CHSH..\n");
  chsh_test();
}


// Gnuplot draw correlations command line: (first define path to created gr_co.txt file)

// define path where is stored file:
// cd 'C:\physic\graph_gnuplot'

// display graph: correlation + states s toggle
// set title "Correlations ((oo+ee)-(oe+eo))/(oo+ee+oe+eo)";set grid;set xrange [0:360];set
yrange [-1:1];set xtics 45;plot "gr_co.txt" using 1:2 with linespoint title "1:
co.","gr_co.txt" using 1:3 with linespoint title "2: s tog."

// display graph: correlation + states s toggle + single + uu + ref cosine
// set title "Correlations ((oo+ee)-(oe+eo))/(oo+ee+oe+eo)";set grid;set xrange [0:360];set
yrange [-1:1];set xtics 45;plot "gr_co.txt" using 1:2 with linespoint title "1:
co.","gr_co.txt" using 1:3 with linespoint title "2: s tog.","gr_co.txt" using 1:4 with
linespoint title "3: single.","gr_co.txt" using 1:5 with linespoint pt 1 title "4: uu",-
cos(2*x*pi/180.0) ls 7 title "4: -cos(2*x)"
```