Polarizer simulation method.

The realistic simulation of the polarizer define two results:

- The output taken by the photon (e or o)
- The time taken for the crossing (which will be a coefficient to be multiplied with a time constant)

This is done in 2 steps:

- 1: Calculation of an amplitude value depending on the hidden variables of the photon and the angle of the polarizer.
- 2: A test with two threshold values equal to PI/4 and PI/2.

Step 1:

Calculation of the amplitude for the threshold test.

The photon is modelled using 3 variables noted **p, q, r**

With:

p : Angle of polarization.
q and r : Two other angles used to generate a phase jitter.

The value of these three variables are defined by the source during emission with random values between [0..PI] (and uniform distribution).

The amplitude value, noted **e**, is defined as follows:

By setting:

d = p - a_pol

(p: photon polarization)
(a_pol: angle of the polarizer)

Then e is defined as:

e = d/2 + q/6 + r/12

(q and r: local variables associated with the photon)

The amplitude of this value varies between 0 and 3*PI/4, of which 2/3 depends on the difference in polarization angle of the photon / polarizer angle.

Note: if the d value is negative, the polarizer having a periodic PI period operation, the d + PI value is used.

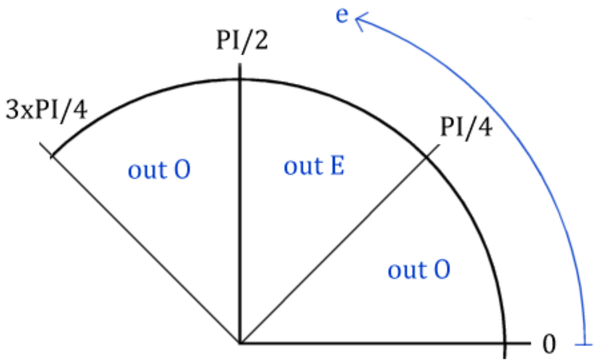
Step 2:

Determination of the output.

It is done with a test comparing the value e with two thresholds values PI/4 and PI/2

If e between PI/4 and PI/2, the output is e, otherwise it is o. (note: this choice is arbitrary).

Here is an explanatory graph:



Coded in C language, this produces the following code:

```
#define OUT_O 0                     // value used to code o out
#define OUT_E 1                     // value used to code e out

int out;
float d, e;

d = pho.p - a_pol;                  // a_pol is polarizer angle
if (d < 0)
  d = d + PI;                       // use positive modulus PI value

e = d/2 + pho.q/6 + pho.r/12;

if ((e >= PI/4) && (e < PI/2))
  out = OUT_E;                      // e in PI/4..PI/2 range
else
  out = OUT_O;
```

Note: pho.p, pho.q, pho.r represent the variables p, q, r associated with the photon.

Calculation of the polarizer crossing time.

When it leaves the polarizer, the polarization of the photon is adjusted to the e or o output.

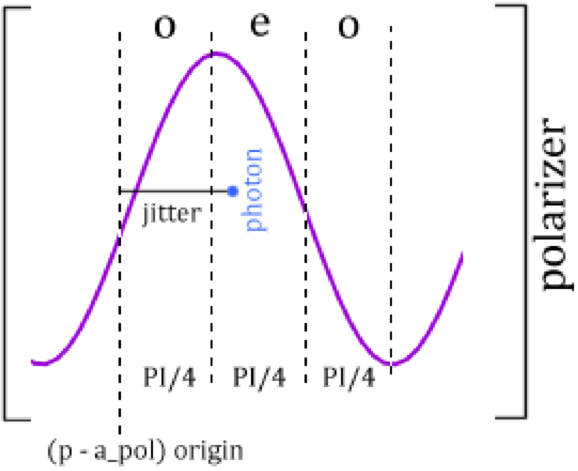
A transit time can be defined by calculating the variation in polarization occurred between its input and output from the polarizer.
The crossing time is then proportional to this repolarization value. (To be multiplied by a constant)
This repolarization can reach a maximum of PI / 2.

Note1: Using a nonlinear, sigmoid-shaped time with an inflection point in PI/4, allow to produce detection correlations in $\sin^2$ if the size of the pairing window is not enlarged. (Setting 'st1 delay' in the simulation program).

Interpretation:

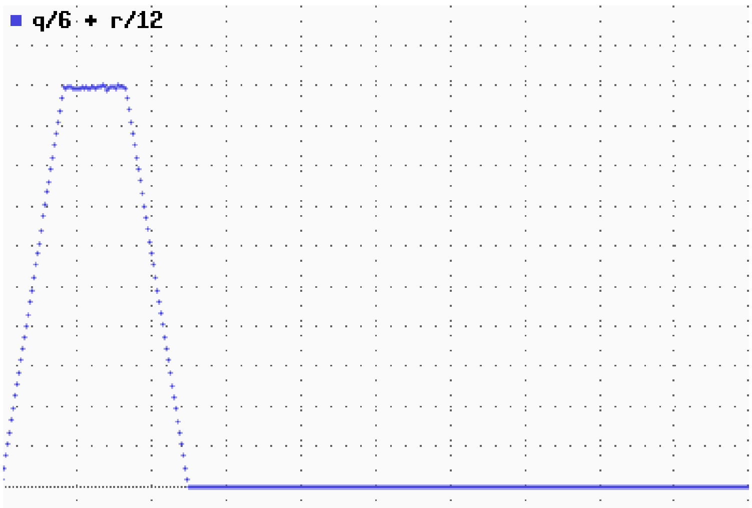
It is difficult to know if this algorithm is only computational, or if it can represent something existing.
In this case, it is necessary to succeed in relating it to elements of a local realistic theory.

A possible personal interpretation is that the sum e represents a phase proportional to a position of the particle. (Which would therefore be locally defined). When interacting with the polarizer, this phase would then precisely define a PI/4 sector associated with an e or o output.



Jitter = Distribution of the sum q/6 + r/12.

Distribution graph (2*q + r)/12



X axis : 0..PI

Note: Use of this algorithm is made on the following web page, describing methods for drawing circles.

http://pierrel5.free.fr/physique/circle/circles.htm (see "Method using randomness and correlations")

Conclusion:

The main objective of this simulation is to show that a local method can still explain the last EPR experiments carried out and that it is still not mandatory to invoke non-locality to explain the results.

Regarding the 'old' experiments using continuous sources and moving window pairing, local simulation also makes it possible to obtain a stable violation of the Eberhard (or CH) inequality with intensities greater than that obtained in these experiments.

Regarding the more recent experiments using fixed windows and pulsed sources, these ones obtained inequality violation intensities with a low amplitude and very close to 0.

The simulation, using the positivity test tool, shows that very low amplitudes are very dependent on stochastic variations.
An experiment would then have to be repeated a large number of times (10 to 20 times), using a large number of samples to validate the stability of a result.
It also shows, by simulating QM, that higher amplitudes would have to be obtained if the non-locality applied.

The simulation also makes it possible to show that with a local model, a stable violation can be obtained if the multiple detections (accidental) are ignored and counted as uu. (No detection on Alice and Bob).
Even doing a normalization on the number of measurements, it can affect the sign of the inequality.

If the multiple detections are not produced by randomness, but result from multiple emissions of pairs by the source, restarting the measurement by counting it as uu indirectly allows to filter the situations producing uu measurements, corresponding to specific conditions not generating correlations, replacing them with detection of the second emitted pair, without this being detectable.
This ultimately generates a counting error for the result of the inequality.
It is therefore necessary, when counting, to take into account all the multiple measurements, and to count each detection as a single measurement.

However this last point is not described in the results of experiments, seeming to be secondary.
It is however crucial, because even with a low rate of incorrectly counted measurements (2%), a stable violation of inequalities can also be produced with a local system and pairing with fixed windows.