Refraction law and Fermat principle: a project using the ant colony optimization algorithm for undergraduate students in physics

Q L Vuong , C Rigaut and Y Gossuin
UMONS. Place du Parc, 20. 7000 Mons, Belgium
E-mail: quoclam.vuong@umons.ac.be

Abstract

A programming project for undergraduate students in physics is proposed in this work. Its goal is to check the Snell–Descartes law of refraction using the Fermat principle and the ant colony optimization algorithm. The project involves basic mathematics and physics and is adapted to students with basic programming skills. More advanced tools can be used (but are not mandatory) as parallelization or object-oriented programming, which makes the project also suitable for more experienced students. We propose two tests to validate the program. Our algorithm is able to find solutions which are close to the theoretical predictions. Two quantities are defined to study its convergence and the quality of the solutions. It is also shown that the choice of the values of the simulation parameters is important to efficiently obtain precise results.

Keywords: ant colony optimization, programming project, refraction, Fermat principle, physics simulation

1 Introduction

For almost 40 years, computers have been introduced into schools to help students learning science, with more or less success. It has been shown that computers can improve their performance compared to traditional instruction[1]. In physics, computers have considerably changed the everyday life of the researchers, allowing them to process experimental data more quickly, solve equations which do not have analytical solutions or simulate very complex systems. More specifically, computing has become essential in molecular dynamics, statistical physics, fluid dynamics, quantum physics, etc[2]. The introduction of programming in physics courses seems essential nowadays to most physicists[3, 4] and helps graduated students to find jobs or do their research in a more efficient way [2, 5, 6]. It also provides new ways to understand physics[5] and new motivation for students who seem more involved when they can 'numerically test' the theories[7]. It is thus natural that computing has been introduced into physics curricula using several approaches[5, 8]:

- Built-in software programs can illustrate physical phenomena using graphics or 3D animation and help students to understand it in a more intuitive way[9]

- Independent programming courses (which can be common with mathematics or computer sciences students)
- Solving physics exercises by programming
- Computational physics courses given in advanced Master or Undergraduate programs
- Creation of Majors/Masters in Computational Physics.

In our Physics Department, a pure programming course has been introduced in the second year of undergraduate courses. The next year, students follow a numerical analysis course and a small simulation course. A student project week is also organized during this year: our research unit often proposed small coding projects, which allow students to apply their programming skills in physics problems for the first time. Indeed, it has been proved that one of the most efficient ways to improve programming skills and student motivation is to use a project-based approach[10]. Such programming projects are similar to research projects in the way that they are open-ended problems with no straight answer, but have the disadvantage of being very time-consuming[11]. Our students write the codes from scratch using the smallest number of libraries (only random number generator libraries are used), which forces them to understand all the details of their algorithm without using any 'black boxes'[4]. The project presented in this article was proposed to one student during this project week.

A lot of physics problems can be formulated as optimization problems: mechanics with the principle of least action, equilibrium states, which minimizes the energy of the studied system, general relativity which constrains light to follow the geodesics of space-time, etc. Optimization algorithms are thus very useful tools for the physicist. Classical optimization algorithms are not always suitable for complex problems as they require computation of all the possible configurations. The processing time of these algorithms exponentially increases with the system size. Other optimization algorithms introduce a random factor in the search for the optimal solution, which increases the performance of the computation.

The ant colony optimization (ACO) algorithm was originally proposed as an optim- ization algorithm to solve the traveling salesman problem. Based on the behavior of an ant colony searching for the shortest path between its nest and a source of food, this method presents several advantages from a pedagogical point of view: it is intuitive and easily understood, easy to program and to parallelize, does not require advanced mathematical knowledge and uses random walk to converge to the solution. The structure of the algorithm encourages the use of object-oriented programming.

In this work, we present a project which can be proposed to undergraduate students in physics, which consists of applying the ACO to the Fermat principle—more specifically, to refraction—in classical optics. Fermat principle postulates that the light path connecting two points is the path that takes the least time[12]. In other words, the Fermat principle postulates that light always minimizes its optical path. A consequence of this principle is that light is refracted when it strikes the interface of two media characterized by different refractive indexes $n_i = c/v_i$ where c is the speed of light in the vacuum and v_i is the speed of light in the medium i. This phenomenon is described by the Snell–Descartes law

$$n_1 \sin\theta_1 = n_2 \sin\theta_2 \tag{1}$$

where θ_1 and θ_2 are the incidence and refraction angles (figure 1). This law can easily be derived from the Fermat principle by minimizing the light duration from a defined initial position and

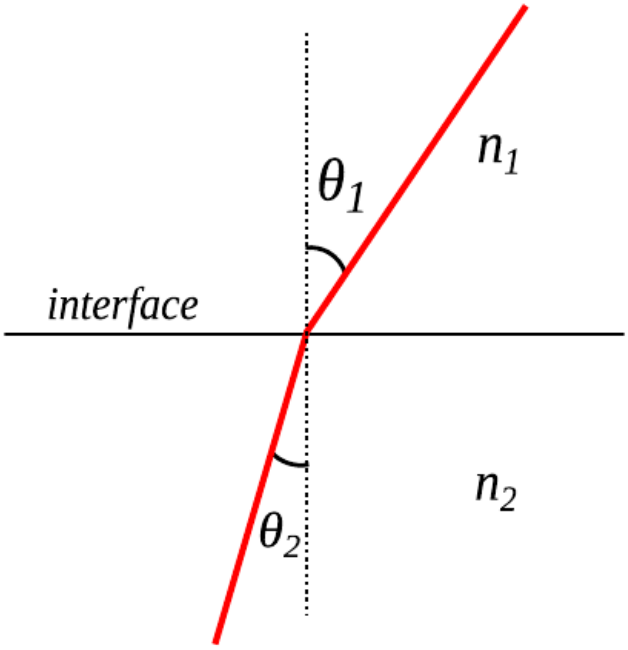

Figure 1: Refraction of light between two media with refractive indexes n_1 and n_2.

a defined final position. The goal of the proposed project is to use the ACO to minimize the light duration path between an initial and final positions. The obtained numerical solution can then be compared to the prediction 1. Interestingly, the idea to check the Snell–Descartes law on real ants has already been carried out[13]: it has been observed that when the ants cross two media which modify their speed, a 'refraction' of their path occurs and obeys the Snell–Descartes law.

Methods and algorithm

An ant colony can find the optimal path to a source of food thanks to the pheromones that each ant leaves on its way. Initially, the ants walk randomly to find food and drop pheromones on their way. The more concentrated at one position the pheromones are, the more the ants tend to choose the corresponding position. As the time to make a return trip on the shortest path for an ant is lower than all the others, this optimal path will be characterized by a more concentrated quantity of pheromones than any other paths. Thus, after a while, the optimal path contains the greatest amount of pheromones and most of the ants of the colony follow it.

The ACO algorithm is based on this ant collective behavior to find the optimal solution of a problem[14]. It was introduced during the 1990s by Dorigo *et al*[15] to solve the traveling salesman problem: in this problem, a salesman must find the shortest route that links all the cities he has to visit. The ACO belongs to the class of the swarm intelligence algorithms in which 'agents' must cooperate to find optimal solutions—mimicking the collective behavior of some natural species such as birds, bees or ants[16, 17]. Although quite new, this algorithm has been mathematically proved to converge[18] and can be applied to a wide variety of problems such has scheduling elaboration, geometrical optimization, network communication, protein conformation, etc.

The ACO algorithm used in this work is inspired by the one introduced by Dorigo et al[15] and the one by Angus[19]. Pechac has also used the ACO algorithm for light travelling[20] in more complex systems. The ACO algorithm was adapted for our simple refraction problem and is composed of the following steps:

1. Our rectangular simulation space is composed of $n_x \times n_y$ nodes (i, j) homogeneously distributed along the x (horizontal) and y-axis (vertical) and is divided into two media: every node above (resp. below) y_{int} is characterized by a refractive index n_1 (resp. n_2).

2. At the beginning of the algorithm, each transition $(i, j) \rightarrow (i', j')$ is associated to an initial quantity of pheromones $\tau_{i,j\rightarrow i',j'}$. If a transition is forbidden, the initial quantity of pheromones is set to zero.

3. For each walking session, n A ants choose their path:

   (a) For each ant:

      i. The ant is initialized at the defined initial position (i_{init}, j_{init})

      ii. Until the ant reaches the defined final position (i_{final}, j_{final}): The new position (i', j') of the ant is chosen with a probability $p_{i,j\rightarrow i',j'} \propto \tau_{i,j\rightarrow i',j'}$ where (i, j) is the current position and $\tau_{i,j\rightarrow i',j'}$ is the pheromone associated to the transition $(i, j) \rightarrow (i', j')$. The ant is not allowed to leave the simulation space: reflecting boundaries are imposed.

   (b) At the end, the path duration of each ant is calculated thanks to the refraction indexes. The duration of a whole path is the addition of the duration of each transition that composes the path. The duration of a transition $(i, j) \rightarrow (i', j')$ in a medium of refractive index n (normalized by c) is given by $n\sqrt{(i - i')^2 + (j - j')^2}$.

   (c) If the duration of the optimal path of the walking session is smaller than the one of the previous session, it is saved.

   (d) All the pheromones are evaporated by updating each value $\tau_{i,j\rightarrow i',j'} \leftarrow (1-\rho)\tau_{i,j\rightarrow i',j'}$ where ρ is a positive evaporation constant lower than 1.

   (e) The pheromones are updated considering all the ant paths achieved during the walking session. If a transition $(i, j) \rightarrow (i', j')$ occurred for a path characterized by a duration λ, the associated pheromone $\tau_{i,j\rightarrow i',j'} \leftarrow (1 - \rho)\tau_{i,j\rightarrow i',j'}$ is updated according to the rule $\tau_{i,j\rightarrow i',j'} \leftarrow \tau_{i,j\rightarrow i',j'} + \frac{Q}{\lambda}$ where Q is a simulation parameter defined by the user.

When not explicitly mentioned, the default values used for the parameters are

- $n_A = n_x \times n_y$
- Number of walking sessions: 50 000
- $\rho = 0.02$
- $Q = \frac{(n_1+n_2)(n_x+n_y)}{2n_A}$
- Initial pheromone value for each transition: $\frac{1}{(n_1+n_2)(n_x+n_y)}$

- An ant is only allowed to update its position up to 2 units in each x, y-axis i.e. $(x(t+\Delta t), y(t+\Delta t)) = (x(t)+\Delta x, y(t)+\Delta y)$ with $|\Delta x| \leq 2$ and $|\Delta y| \leq 2$. Care has to be taken when an ant crosses the medium interface: the associated duration has two contributions — one from the medium n_1 and one from the medium n_2.
- The simulation space was a square grid composed of 10×10 nodes.
- The initial position is taken at the bottom left of the grid (coordinate (0,0)) and the final position is taken at the top right of the grid (coordinate (9,9)).

These values, which result from a compromise between the precision of the computed results and the processing time, are discussed further in section 3.3. Our program was written in C++, used the GNU Scientific library and was parallelized in OpenMP.

2 Results and discussion

2.1 Validation

As for every program, a testing phase is necessary to ensure its validity. It can be interesting to let students elaborate their own tests. In our work, the program was tested by considering the case $n_1 = n_2$. In that case, the optimal solution is a straight line between the initial and final position. Our results followed the theoretical predictions with a coefficient of determination $R^2 = 1$.

Another interesting way to check the program is to evaluate the pheromone value when all the ants converge to the same path. Indeed, in this case, the pheromone associated to one transition is given by

$$\tau(t + \Delta t) = (1 - \rho)\tau(t) + n_A \left(\frac{Q}{\lambda}\right)$$

When the algorithm converges (i.e. when most of the ants of the walking session choose the same path), the transition pheromones of the path followed by all the ants take a stationary value τ^* given by

$$\tau^* = (1 - \rho)\tau^* + n_A \left(\frac{Q}{\lambda}\right)$$

i.e.

$$\tau^* = \frac{n_A}{\rho} \left(\frac{Q}{\lambda}\right)$$

This expression can be easily compared to the values provided by the simulation and should be approximately equal to them at the end of the simulation. The other transitions not taken by the ants should be approximately equal to zero.

2.2 Refraction law

Simulations were performed for different refraction indexes n_2 ranging from 1–5 (and fixing $n_1 = 1$). A typical example is shown in figure 2 where the optimal path found by the algorithm and the theoretical one are shown. Considering the discretization of the simulation space, the obtained solution is qualitatively close to the theoretical one. This optimal solution has been

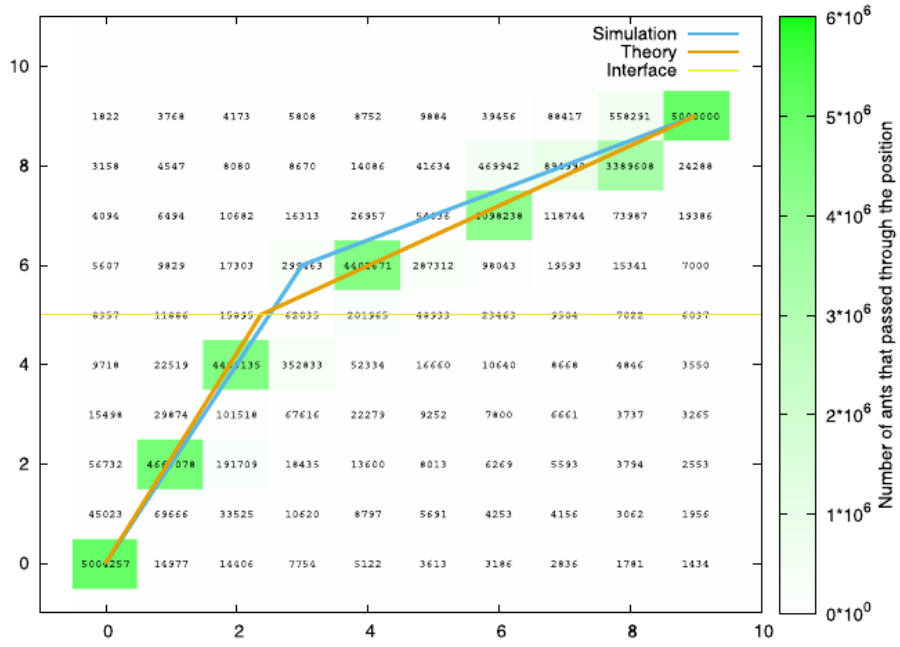

Figure 2: Simulation for $n_1 = 1$ and $n_2 = 2$. The number of times the ants reach a position is also indicated. The optimal trajectory found by the algorithm is good compared to the theoretical one considering the simulation space discretization (only ten nodes on each axis).

found after $100 \times 50000 = 5\ 10^6$ calculated paths, which should be compared to the $100! \approx 10^{158}$ paths that should be computed by a naïve algorithm.

Several methods can be proposed to students to quantitatively evaluate the quality of the numerical solutions: for example, computing the coefficient of determination R^2 of the solution, the duration of the path, evaluating the refractive angle and comparing it to the Snell–Descartes law, or evaluating the intersection x cross of the light ray and medium interface.

The coefficient of determination of the simulation results is defined as

$$R^2 = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2}$$

where y_i are the y-coordinates of the optimal path obtained by simulation, $\bar{y}$ their average and $\hat{y}_i$ are the y-coordinates of the theoretical path. Its value characterizes the quality or closeness of the obtained solution compared with the theoretical one and is close to 1 when the two curves are close to each other. Our obtained R^2 were close to 0.99 for all our simulations except for $n_2 = 4$ and 5 with 10×10 grids. In that case, the poor resolution of the simulation space does not allow the ants to follow a line which is too sloping (see figure 3). Indeed, it is a typical problem of numerical resolution: as shown in figure 5, when the ant is allowed to jump a distance $|\Delta x|, |\Delta y| \leq 2$, the minimum angle θ which it can make is 27°. Any line that is more sloping, i.e. with $\theta < 27°$, will be difficult for the ant to reproduce: which is the case when $n = 4$ where this angle must be equal to 24°. Increasing the resolution of the simulation solves the problem (figure 4): for a 20×20 grid with $|\Delta x|, |\Delta y| \leq 4$, $\theta = 14°$ and the coefficient determination is also close to 0.99 for $n_2 = 4$ and 5. Those cases illustrate one of the main disadvantages of numerical tools (the numerical precision) and provide a good opportunity to discuss with students the importance of the numerical parameters of their simulations and other possible alternatives

to solve the Fermat principle: root finding or maximization algorithms as a bisection method, gradient method or shortest-path algorithms such as Dijkstra or Bellman–Ford algorithms.

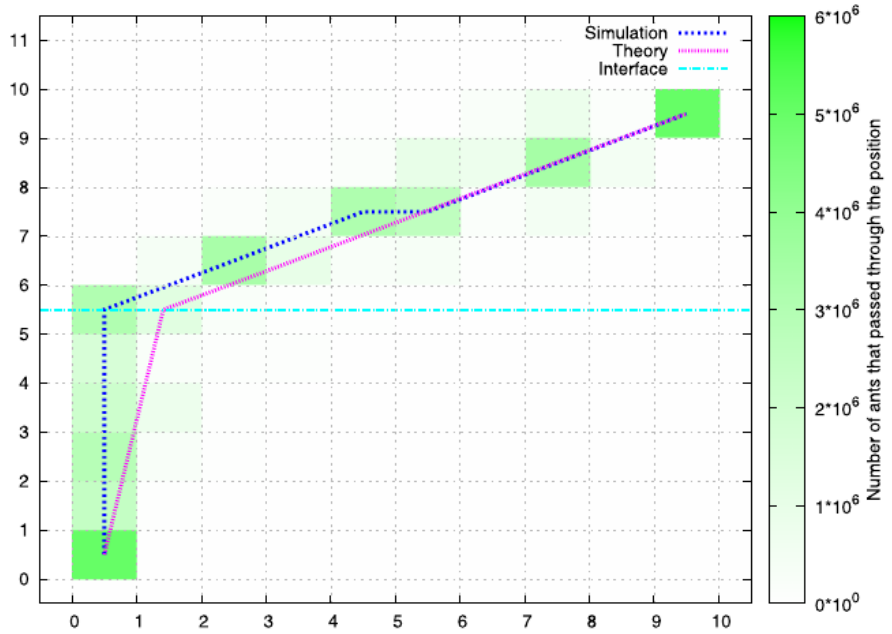

Figure 3: Simulation for $n_1 = 1$ and $n_2 = 5$ with a 10×10 resolution grid using $|\Delta x|, |\Delta y| \leq 2$. The numerical solution is less close to the theoretical one, with a vertical line in the part below the interface. It is confirmed by its $R^2 = 0.46$.

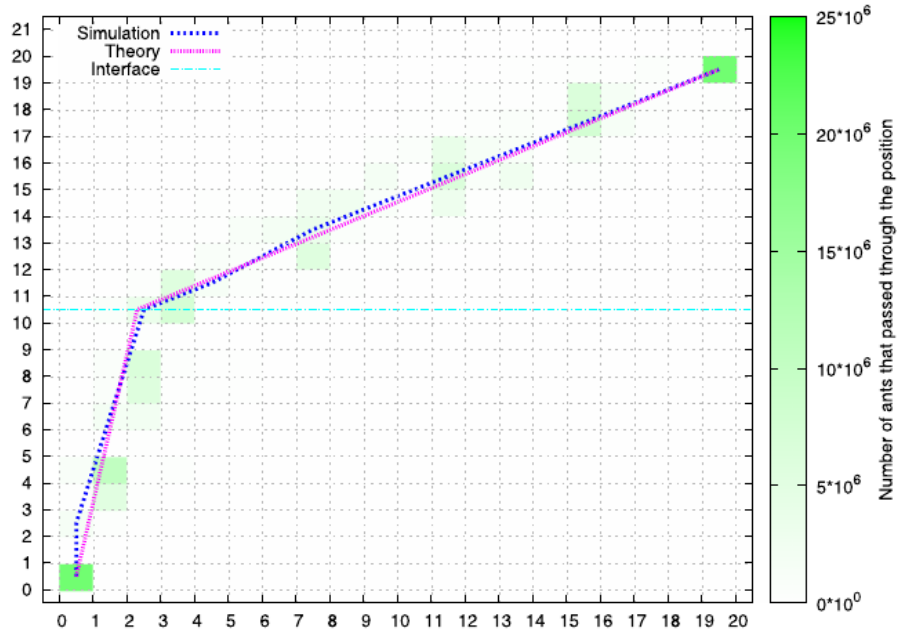

Figure 4: Simulation for $n_1 = 1$ and $n_2 = 5$ with a 20×20 resolution grid using $|\Delta x|, |\Delta y| \leq 4$. With this resolution, the numerical solution is closer to the theoretical one compared to figure 3, characterized by an $R^2 = 0.99$.

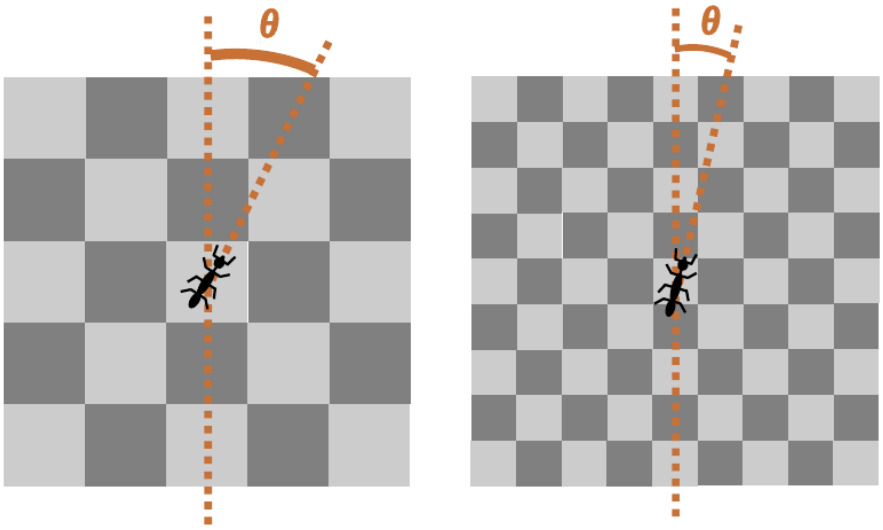

Figure 5: The highest slope (characterized by the θ angle) that can be achieved by an ant depends on the resolution of the grid: with $|\Delta x|, |\Delta y| \leq 2$ (figure on the left), θ is larger than the one corresponding to the $|\Delta x|, |\Delta y| \leq 4$ case.

The intersection (x_{cross}, y_{cross}) was also compared to theoretical predictions using the analytical expression of the light path duration:

$$\begin{aligned} \text{light duration} =& n_2 \sqrt{(i_{init} - xcross)^2 + (j_{init} - ycross)^2} \\ &+ n_1 \sqrt{(i_{final} - xcross)^2 + (j_{final} - ycross)^2} \end{aligned} \tag{2}$$

More precisely, the root of the partial derivative of equation 2 with respect to x_{cross} was numerically computed using the bisection method. Figure 6 shows that the simulation results follow qualitatively the theoretical predictions and are closer to them when the resolution of the grid is taken higher.

2.3 Simulation parameters

It can be interesting for students to study how the simulation results depend on the simulation parameters: it emphasizes the importance of finding the right and optimal simulation parameters to make the search for solutions efficient. It also shows that bad simulation parameters can lead to wrong numerical solutions. A quantitative discussion about the quality of the numerical solutions or the convergence rate of the ant paths requires defining the appropriate coefficients. In our work, we defined two coefficients:

$$\epsilon_{th} = \sqrt{\sum_{\substack{\text{nodes } (i,j) \text{ of} \\ \text{the optimal path}}} (j - j_{th}(i))^2}$$

$$\epsilon_{spread} = \sqrt{\sum_{\text{nodes } (i,j)} (n_A(i,j)(j - \langle j(i) \rangle)^2)} \text{ where } \langle j(i) \rangle = \frac{\sum_j n_{Ai,j} j}{\sum_{i,j} n_{Ai,j}}$$

where $n_A(i,j)$ is the number of ants that passed through the position (i,j), $j_{th}(i)$ is the theoretical path. ϵ_{th} evaluates the quality of the solution by comparing its deviation from the

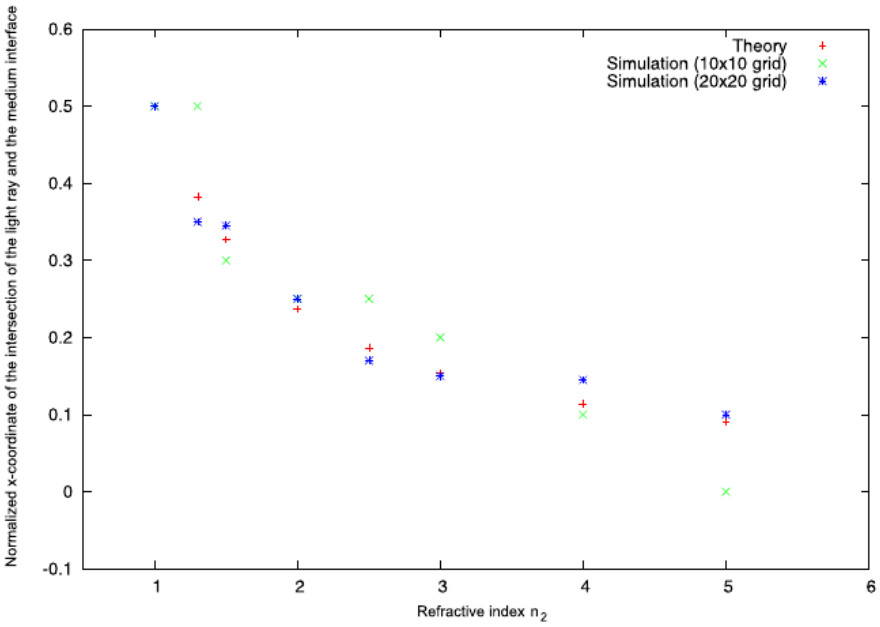

Figure 6: Comparison between the intersection between the light ray and medium interface for two different simulation grid sizes. The grid with the highest resolution predicts the value with more precision than the lowest one, as expected. For 20×20 grids, transitions with Δx and Δy up to four units were accepted.

theoretical one. ϵ_{spread} evaluates the convergence of the ants by computing the 'spread' of their position (compared to their average path) on the simulation grid. For 10×10 grids, a solution with $|\epsilon_{th}| \leq 0.7$ was considered as good and we considered the convergence was good when $|\epsilon_{spread}| \leq 1$.

These coefficients allow us to study the importance of the values of some of the simu- lation parameters: as can be expected, an increase in the number of walking sessions and ants leads to a more precise and better convergence of the solution: decreasing the number of walking sessions leads to ant paths which are more scattered in the simulation space (figure 9). We also observe the same trend for the value of Q even if the variations are less significant.

The value of the evaporation parameter ρ is decisive for the convergence of the solution: if it is too low, all the paths found by the ants will contribute to the pheromones and the algorithm converges slowly to the right solution (figure 8). If it is too high, every walking session only depends on the previous one and the algorithm can converge to a wrong solution (figure **??**).

We also studied the influence of the transition values Δx and Δy: as expected the larger they are, the more precise the proposed solution is (figures 3 and 4). Note that if this value is equal to one, the precision is too low and the solutions found by the simulations can be completely wrong (figure 10).

3 Conclusion

Computers and programming have become inevitable for a physicist and it is therefore important for students in physics to learn these skills during their studies. Programming skills can only evolve by exercising and developing projects. We proposed a simple physics problem,

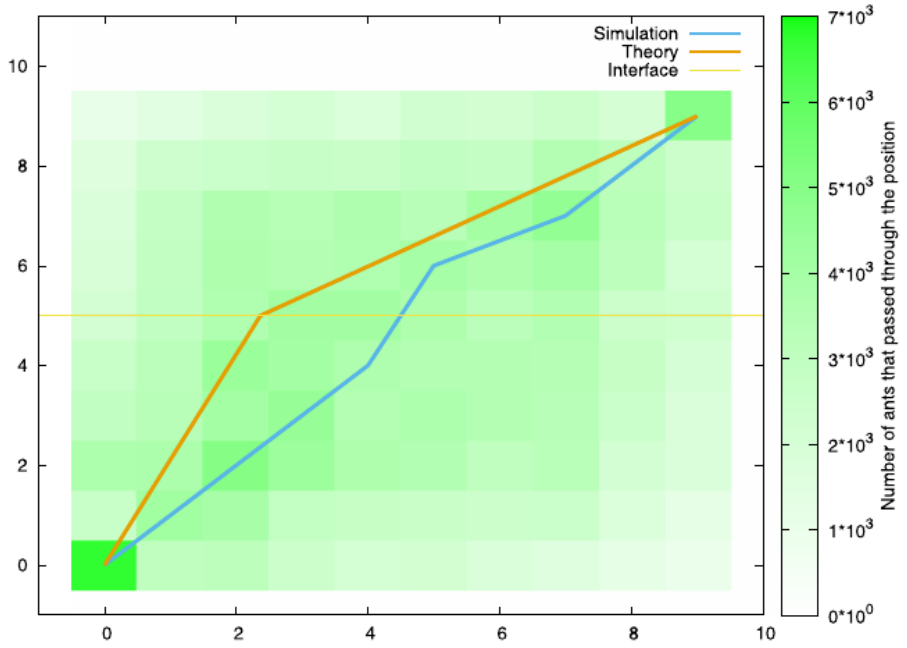

Figure 7: Simulation for $n_1 = 1$ and $n_2 = 2$, with a number of walking sessions equal to 50. Compared to figure 2, ant walks are more scattered in the simulation space and the quality of the optimal solution found by the algorithm is very poor.

which can be numerically solved with different tools that a physicist will encounter in his or her career: random process, parallelization or object-programming. It is shown, as in all modern physics problems, that the solutions found by the algorithm must be interpreted with care by considering how the simulation parameters or the resolution of the simulation space can influence the results.

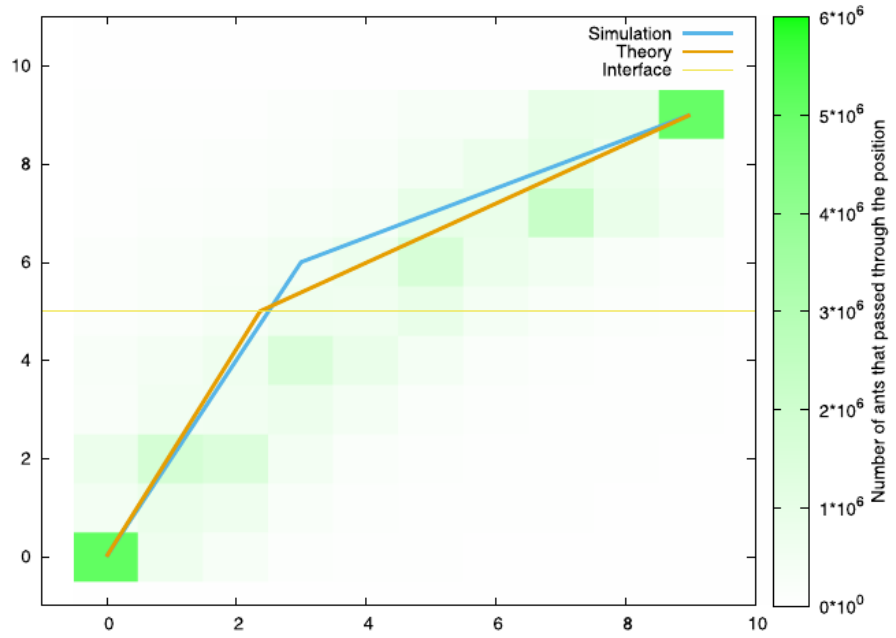

Figure 8: Simulation for $n_1 = 1$ and $n_2 = 2$, with $\rho = 0.0002$. Compared to figure 2, the ants converge more slowly to the optimal solution (dull green squares).

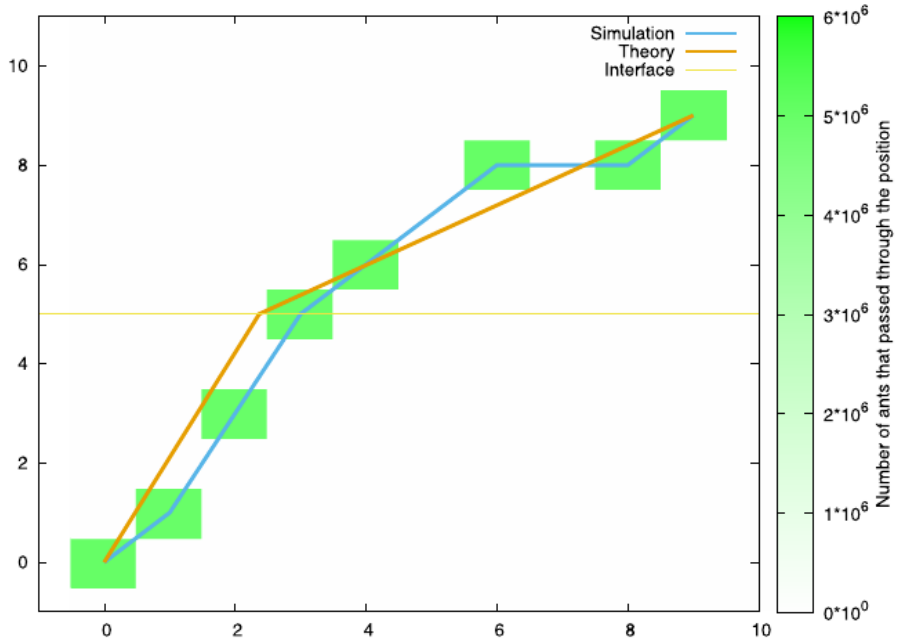

Figure 9: Simulation for $n_1 = 1$ and $n_2 = 2$, with $\rho = 1$. In this case, the ants converge to a solution (intense green squares), which does not correspond to the theoretical optimal solution.

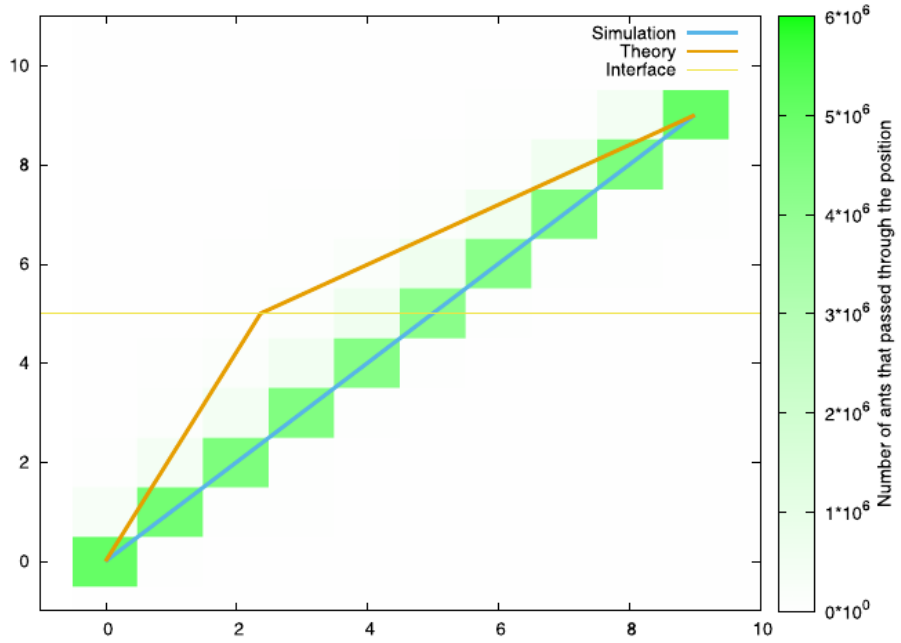

Figure 10: Simulation for $n_1 = 1$ and $n_2 = 2$, with $\Delta, \Delta y = 1$. In this case, the discretization of the ant paths is too poor and a straight line has a smaller optical length than a path that would follow the theoretical one.

References

[1] K. Wise, "The effects of using computer technologies in science instruction: a synthesis of classroom-based research," in *1988 AETS Yearbook* (J. D. Ellis, ed.), pp. 105–18, Colorado Springs, CO: Office of Educational Research and Improvement, 1988.

[2] R. H. Landau, *A survey of computational physics: introductory computational science*. Princeton: Princeton University Press, 2008.

[3] R. Fuller, "Numerical Computations in US Undergraduate Physics Courses," *Computing in Science & Engineering*, vol. 8, no. 5, pp. 16–21, 2006.

[4] R. Landau, "Computational Physics: A Better Model for Physics Education?," *Computing in Science & Engineering*, vol. 8, no. 5, pp. 22–30, 2006.

[5] N. Chonacky and D. Winch, "Integrating computation into the undergraduate curriculum: A vision and guidelines for future developments," *American Journal of Physics*, vol. 76, no. 4, pp. 327–333, 2008.

[6] J. Taylor and B. King, "Using Computational Methods to Reinvigorate an Undergraduate Physics Curriculum," *Computing in Science & Engineering*, vol. 8, no. 5, pp. 38–43, 2006.

[7] K. Aho and K. Chandra, "Introducing programming into the physics curriculum at Haverhill High School using the R language," (Bridgpeort, CT), University of Bridgeport, 2014.

[8] D. M. Cook, "Computation in undergraduate physics: The Lawrence approach," *American Journal of Physics*, vol. 76, no. 4, pp. 321–326, 2008.

[9] F. Esquembre, "Computers in physics education," *Computer Physics Communications*, vol. 147, no. 1-2, pp. 13–18, 2002.

[10] D. Davenport, "Experience using a project-based approach in an introductory programming course," *IEEE Transactions on Education*, vol. 43, no. 4, pp. 443–448, 2000.

[11] H. Gould, "Computational physics and the undergraduate curriculum," *Computer Physics Communications*, vol. 127, no. 1, pp. 6–10, 2000.

[12] R. P. Feynman, M. Sands, and R. B. Leighton, *The Feynman lectures on physics*. Redwood City, California [etc.: Addison-Wesley, 1989.

[13] J. Oettler, V. S. Schmid, N. Zankl, O. Rey, A. Dress, and J. Heinze, "Fermat's Principle of Least Time Predicts Refraction of Ant Trails at Substrate Borders," *PLoS ONE*, vol. 8, no. 3, p. e59739, 2013.

[14] M. Dorigo and T. Stützle, *Ant colony optimization*. Cambridge, Mass: MIT Press, 2004.

[15] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics)*, vol. 26, no. 1, pp. 29–41, 1996.

[16] S. Das, A. Abraham, and A. Konar, *Computational intelligence in bioinformatics*. No. 94 in Studies in computational intelligence, Berlin: Springer, 2008.

[17] C. Blum and D. Merkle, *Swarm intelligence: introduction and applications*. Berlin, Germany: Springer, 2010.

[18] M. Dorigo and C. Blum, "Ant colony optimization theory: A survey," *Theoretical Computer Science*, vol. 344, no. 2-3, pp. 243–278, 2005.

[19] D. Angus, "Solving a unique Shortest Path problem using Ant Colony Optimisation," 2010.

[20] P. Pechac, "Electromagnetic wave propagation modeling using the ant colony optimization algorithm," *Radioengineering-Prague*, vol. 11, 2002.