



# High resolution calculation for the optimisation of commercial LED spectra in health applications

Adrià Huguet-Ferran<sup>\*a,b</sup>, Cristian Morales Pérez<sup>c</sup>

*a Kumux, Enlighting Technologies SL, Baldori Reixac 4, Parc Científic de Barcelona, Torre R, Planta 2, B1, 08028 Barcelona, Spain*

*b Departament d'Enginyeries: Electrònica, Universitat de Barcelona, Facultat de Física, Martí i Franquès 1, 08028 Barcelona, Spain*

*c Barcelona Supercomputing Center (BSC), Barcelona, Spain*

---

## Abstract

The proposed project is related to the installation of bioadapted lamps at a hospital facility. The different partners involved in the project are: the scientific team in charge of the hospital itself, a manufacturer of light sources, a lamp control platform and Kumux - Enlighting Technologies (ET). In this context, ET will be responsible for performing the high-resolution calculations of the LED system in the lamp to be able to choose different light scenarios taking into account the visual, non-visual and power properties of light, such as luminous and radiant flux or the energy efficiency of the particular light scenario. The main goal of the project is, once the results of the simulation have been processed and replicated to the lamps, to be able to quantitatively demonstrate the incidence of bioadapted light in the recovery of patients, since there are preliminary studies that show very significant reductions in the hospitalisation period of the patient.

With the computer means available in Kumux, it is not possible to perform high-resolution calculations of the lamps used in the project. Our computational system only allows calculations at low resolution, so in this SHAPE project, we aim to simulate a lighting scenario at high resolution, which would take months of computation with our current code and computer means.

---

## Scientific context

Light is the agent that allows us to see and makes things visible. Within our eye there are a kind of photoreceptor cells, called cones and rods, which are responsible for perceiving the colour and light intensity that enters to the retina. Throughout history, these photoreceptors have evolved to adapt to our environment, illuminated until very recently (in evolutionary terms) by the Sun. For this reason, sunlight has ideal colorimetric properties for the human eye, as our eye has evolved to distinguish the maximum number of colours and shades under this type of lighting.

In 2001, a group of biologists discovered that, inside our eye, there are other types of receptor cells called Intrinsically Photosensitive Retinal Ganglion Cells (ipRGCs). These cells, unlike cones and rods, are related to the non-visual effects of light [1]. Scientifically, they have been shown to play an important role in synchronising circadian rhythms, providing information about the length of day and night, and contributing to the regulation of melatonin, a hormone related to the sleep-wake cycle. These photoreceptors have the peak absorption around 480 nm, corresponding to the blue colour of the visible spectrum [2]. Evolutionarily, our ancestors were exposed to a dynamic natural illumination, with a very important variation of the blue content in the spectral distributions that enter through the retina.

---

\* Corresponding author email address: [adria.huguet@kumux.io](mailto:adria.huguet@kumux.io)

During the day, outdoors, they were exposed to a very important blue content while during the night this blue content was practically imperceptible. For this reason, it is important to design dynamic lighting systems that have consequences beyond the colorimetric properties for the correct synchronisation of circadian rhythms.

Until a few years ago, the technology available on the market to create lighting solutions was very limited. The market was dominated by technologies (incandescent, fluorescent, sodium, etc.) that allowed no possibility of spectral control. However, with the emergence of LEDs, a very large number of possibilities arose, and they are still under investigation. These include the design of dynamical spectral lighting systems to meet human needs, either at a colorimetric level or at a non-visual level. Lamp manufacturers face many problems in providing such solutions, as the general lighting industry has a very poor scientific base. For this reason, a very important interaction is currently emerging between companies providing specific scientific and technological solutions and lamp manufacturers.

Using a supercomputer to run and test algorithms provides an important competitive advantage, since in many cases the algorithms have very intensive calculation needs considering the non-linearity of the results and the hard implementation of classic optimisation algorithms. The output and results of the algorithm will be included into the prototypes of the industrial partner in order to be measured experimentally and corrected if necessary. Usually, there is a small difference between simulations and experimental measurements which, although they are usually imperceptible, must be taken into account when carrying out scientific studies.

The simulation part of this project consists of performing the calculations of a computation algorithm related to the mixture of different spectra of visible light LEDs. The algorithms dedicated to the spectral mixture of light face the problem of metamerism. Metamerism is a psychophysical phenomenon in which infinite samples of spectra can coincide in the same colour. In addition, the concept of metamerism is extensible to all kinds of colorimetric parameters (such as Correlated Colour Temperature (CCT), Colour Rendering Index (CRI), Gamut Area Index (GAI), colour coordinates...), non-visual (Melanopic/Photopic Ratio (M/P ratio), Circadian Stimulus (CS)...) or even electrical (energy efficiency). On a practical level, for example, a specific yellowish illumination can be obtained in many different ways: using monochromatic amber LEDs, using RGB LEDs or using cool white and red LEDs. These three possibilities may be emitting exactly the same colour with totally different spectral distributions. However, on a visual level, if an observer looks directly at the light source, it will appear that there are no differences between them.

This problem grows enormously as the number of LEDs of different colours increase. Theoretically, in a system of 4 different LEDs (channels), there are infinite spectra that can generate the same colour because the results are multivalued.

The vast majority of LED electrical current controllers have a resolution ranging from 100 to 4096 steps, which means that it is possible to have from 100 to 4096 light intensities per each LED channel. Therefore, the total number of light combinations that can be emitted in a lamp depends on the resolution of the driver and the number of different LED channels. The number of possible combinations between LED channels follows the relationship:

$$n = r^c \quad (1)$$

where  $n$  is the number of possible combinations,  $r$  is the resolution and  $c$  is the number of channels.

The resolution and the number of channels are very important when calculating the computational resources needed to perform the calculation. For each simulated light combination, the summed spectrum of the different LED channels and the parameters related to colorimetry, non-visual properties and energy of each spectrum are calculated. This process has a computational cost that is not depreciable, especially when there are a large number of light combinations generated. Following Equation (1), if we use a 16-step resolution with 6 different channels,  $16 \cdot 10^6$  different combinations are generated. On an Intel Core i7-6700K processor (4 GHz, 4 cores, 8 threads), paralleling the calculations to take advantage of the computing power of all cores, 13.2 hours are needed to generate the table of results. To increase the resolution to 32 steps to check if the system is avoiding local minima, 35.2 days of computation are needed. Currently, Kumux's computer equipment allows working at a moderate resolution (16). Using a supercomputer, it will be possible to increase the resolution up to 32. Since the majority of LED electrical current controllers have a resolution ranging from 100 to 4096 steps, at the moment it is not possible to work with so much resolution using a 6 channel LED system, even for a supercomputer.

In addition to the multivaluation of the results, it must be taken into account that the colorimetric and non-visual calculation parameters follow non-linear relations with the intensity of the different channels. To better understand this problem, let us take an example with two different white LEDs, one of warm hue with CRI 80 and the other of cool hue with CRI 90. Depending on the spectra of the two LEDs, combining their light, it is possible that, during the transition from warm white to cool white, the CRI reaches intermediate values between 80 or 90 or that at some points it is even higher than 90. The CRI value during the transition cannot be known in advance nor can be interpolated, and it is necessary to perform precise spectral calculations to know exactly the CRI value at each point of the transition. If other different LEDs are also added and other colorimetric parameters such as IES TM-30-18 Rf are calculated, the complexity of the problem increases enormously. If a classical optimisation algorithm for CRI maximisation is implemented, such as gradient descent, it is very easy to fall into a local minimum/maximum since the relationships between the simulated spectra and the calculation of these parameters do not follow linear relationships. Local maxima can be found continuously depending on the initial conditions.

Adding to this problem, it must be taken into account that the industry has the need to optimise more than one parameter at a time. For Human Centric Lighting solutions, for example, it is necessary to consider the colorimetric affectation and the non-visual affectation. Thus, it is necessary to optimise the parameters CRI and CS, with the possibility of modulating the CCT and maximizing the energy efficiency. The lamp will include results with very different light characteristics (combinations with warm hues, cold hues, very high CRI, very high or very low M/P ratio, very high energy efficiency...), so we are interested in finding general values and then being able to filter the results according to the interests of the hospital's scientific team.

Working on the optimisation of several parameters at the same time, for example, maximising the CRI and minimising the CS for a fixed CCT, evident differences can be observed between the spectra calculated in a resolution of 8 steps and 16 steps. In the 8-step resolution, the spectral solution found does not have some components that the 16-step solution has. For example, in a 128-step current controller we can work at a lower resolution to spend less time in the calculations of the simulation. We can work at resolution 4, which implies 32 value steps in the current controller ( $32 \times 4 = 128$ ). If we decide to work at resolution 8, the steps in the current controller will be of value 16, so working at resolution 8 implies that the values of resolution 4 are also being calculated. As expected, the 16-step resolution achieves better results since it contains the 8 resolution results.

For example, if we compare the results of the maximisation of the IES TM-30-18 Rf parameter in a 10000 K CCT at different resolutions, we obtain the Figure 1.

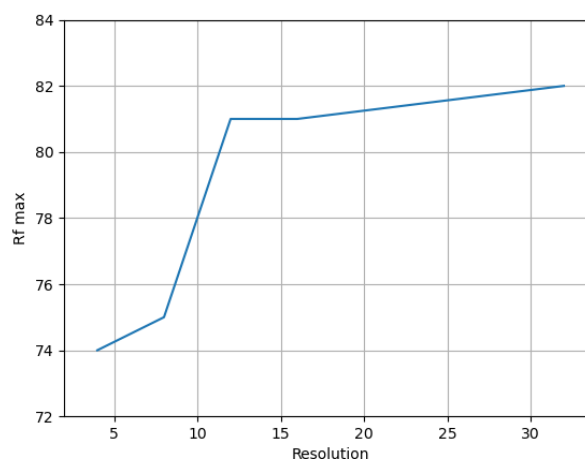


Figure 1: Results of the maximisation of the IES TM-30-18 Rf for 10000 K CCTs at different resolutions.

Another interesting example is the comparison of the maximisation of the M/P ratio parameter around a CCT of 3000 K white light ( $-0.001 < Duv < 0.001$ ), as can be seen in Figure 2.

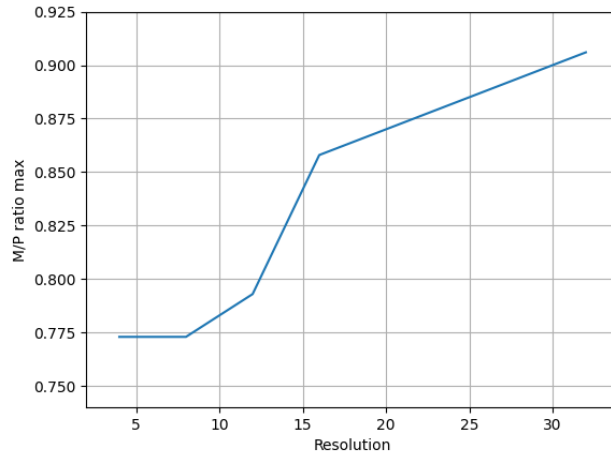


Figure 2: Results of the maximisation of the M/P ratio for 3000 K CCTs white light ( $-0.001 < \text{Duv} < 0.001$ ) at different resolutions.

In both examples it is possible to observe that, as the resolution increases, the value of the optimised parameters also increases. To know the LED system limits of the two optimised parameters (IES TM-30-18 Rf and M/P ratio) it would be necessary to launch a simulation at the limit of the current controller resolution. However, as we have shown previously, due to the high resolution at which they can work, it is not possible to simulate it with the current computational tools.

### Initial version of the code

The code was written in Python 3 and the early versions of the code (version 1.0) used the multiprocessing module as parallelisation strategy. The “multiprocessing” Python package supports spawning processes using an API similar to the threading module. Table 1 shows that the scalability of the version 1.0 of the code was poorly efficient. A first profiling study of the code with cProfile and pySpy showed that there were two main parts of the code that consume most of the execution time. First of all, the simulation runs a part of the code called "Target Engine", which was not parallelised initially and it calculates results depending of a list of parameters given by the input. When the Target Engine ends, the simulation starts the Brute Force part, which was parallelised using multiprocessing and was independent of the Target Engine part. The Brute Force part calculates all the possible combinations by brute force. These combinations can be multiple independent tasks that can be easily distributed among processes.

On Figure 3 we can observe a flame graph generated by pySpy. The first column represents the part of the code Target Engine and the following eight columns represents 8 parallel processes running the Brute Force calculations. In the case of Figure 3, the Target Engine part took 160 seconds and the Brute Force part with 8 process in parallel took 394 seconds.

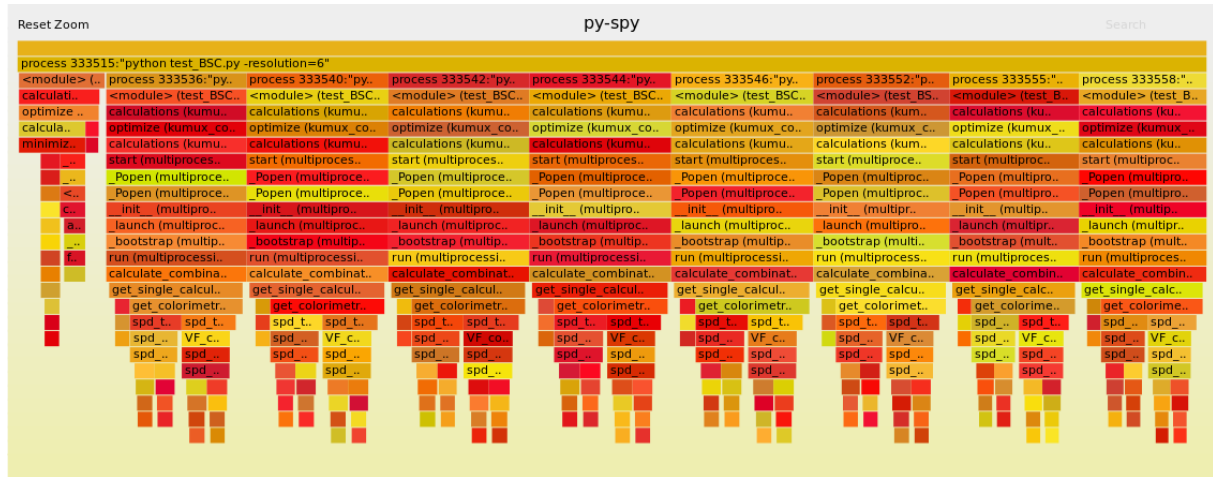


Figure 3: pySpy profiling of a simulation using the 1.0 version with 8 process and resolution 6

On other hand, the multiprocessing library only permits to run on a single node and the code was limited to run up to 48 cores on MareNostrum.

| Cores | Time    | SpeedUp | Efficiency |
|-------|---------|---------|------------|
| 1     | 2969.02 | 1.00    | 100.00%    |
| 2     | 1495.78 | 1.98    | 99.25%     |
| 4     | 846.23  | 3.51    | 87.71%     |
| 8     | 555.18  | 5.35    | 66.85%     |
| 16    | 371.64  | 7.99    | 49.93%     |
| 32    | 287.31  | 10.33   | 32.29%     |
| 48    | 259.22  | 11.45   | 23.86%     |

Table 1: Performance of the version 1.0 of the code with resolution 6

## Enabling for Tier-0

After analysing the initial version of the code, a new version of the code is developed (version 2.0) in order to avoid the main issues commented on in the previous section. In version 2.0, the Brute Force part initially parallelised with *multiprocessing* is replaced with a mpi4py parallelisation in order to enable the code to multi-node simulations. The mpi4py package provides Python bindings for the Message Passing Interface (MPI) standard. Additionally, in version 2.0 of the code, the Target Engine part is also parallelised using the mpi4py packages improving the total execution time. The Target Engine is parallelised dividing each parameter of the input into the MPI processes available for this task. The Brute Force part is parallelised with mpi4py, distributing all the possible combinations over all the MPI processes dedicated to this part of the simulation. At the end of the parallel region of both parts of the code, all the results are gathered and processed to generate the final results and the output. As both parts are independent, they can run simultaneously partitioning the ranks between both parts. To adjust the partitioning a new parameter is included to decide how many MPI processes are involved in each part of the simulation. The imbalance between two the two parts of the simulation can be reduced tuning this new parameter.

| Cores | Version 1.0 Time | Version 1.0 SpeedUp | Version 1.0 Efficiency | Version 2.0 Time | Version 2.0 SpeedUp | Version 2.0 Efficiency |
|-------|------------------|---------------------|------------------------|------------------|---------------------|------------------------|
| 1     | 2969,02          | 1                   | 100,00 %               | 2.745,34         | 1                   | 100,00 %               |
| 8     | 555,18           | 5,35                | 66,85 %                | 345,51           | 7,95                | 99,32 %                |
| 48    | 259,22           | 11,45               | 23,86 %                | 82,78            | 33,16               | 69,09 %                |
| 192   |                  |                     |                        | 21,89            | 125,42              | 65,32 %                |

Table 2: Performance of the versions 1.0 and 2.0 with resolution 6

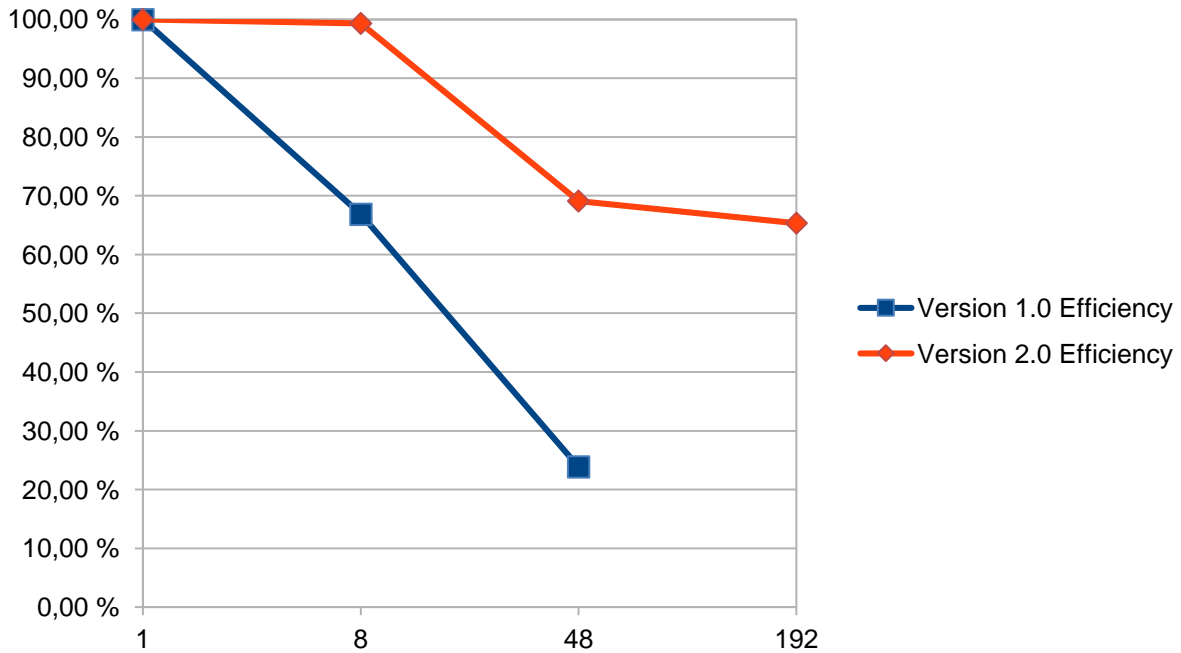


Figure 4: Performance comparison between code versions 1.0 and 2.0

As observed in Table 2 and Figure 4, the performance of the version 2.0 has been improved parallelising the Target engine part and mpi4py enables the code to run on multiple nodes, for example with 192 cores, 4 nodes on MareNostrum. As version 1.0 can not run on multiple nodes, the largest jobs with that version are up to 48 cores, the cores from one node of MareNostrum.

With version 2.0 we were able to run with larger resolutions in larger simulations, but as the number of combinations and the number of results depends on the resolution of the simulation, the generation of all the possible combinations and the output generations were a bottleneck. On version 2.0 (inherited from 1.0), the master process generated all the possible combination for the Brute Force calculations and it stored them on a text file. Then, each process read from this file to get which subsection of the combinations they need to calculate. Additionally, when each process ended its calculations, they sent their local results to the master process. And finally, the master MPI task processed these results and generated an output file with them. In order to enable the code for larger resolutions, we have developed a new version (version 3.0) where the generation of combinations has been adapted to be done on each process involved on the Brute Force part. Therefore, each process computing the Brute Force calculations will generate its own combinations. Additionally, on the version 3.0 of the code, when a process ends its calculations, it processes its own results and then it writes them in on a CSV file.

For example, as observed on Figure 5, using the version 2.0 with a simulation with resolution 16 using 4800 cores, the calculations take 280 seconds but the input and output generation on the master process takes more than 7600 seconds. Using the new implementation, the input and output generation is reduced to 120 seconds. The main performance improvement comes from the output generation, because as soon as a process finishes their calculations, it writes them on the output file. On previous versions, it is done sequentially (rank by rank) by the master. In the case of a simulation with 9600 cores, with resolution 32 and with the new implementation, it takes

7200 seconds for the calculations and 1100 seconds for the input/output. It has been impossible to run a simulation with a resolution of 32 with the version 1.0 and 2.0 of the code due to a large number of combinations and results generated and gathered on the master node.

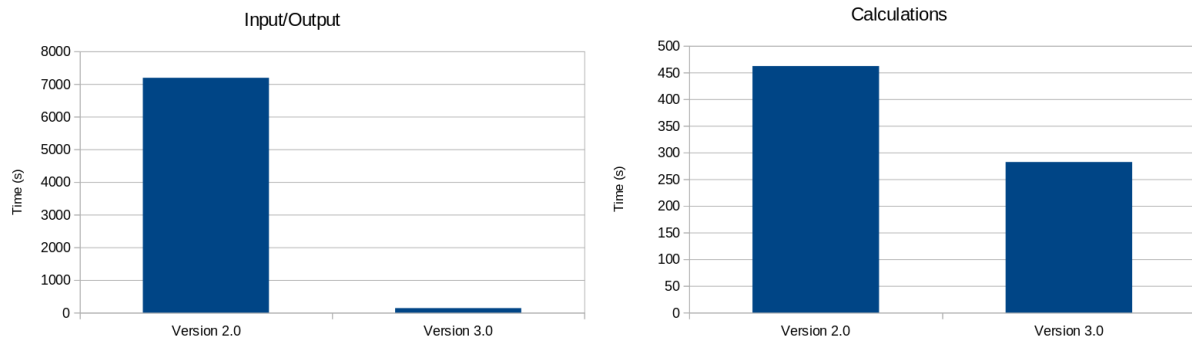


Figure 5: Left: Time spent on input/output of versions 2.0 and 3.0. Right: Time spent on calculations of versions 2.0 and 3.0.

Also, as observed in Figure 6, the scalability of the version 3.0 has been improved compared with the previous versions.

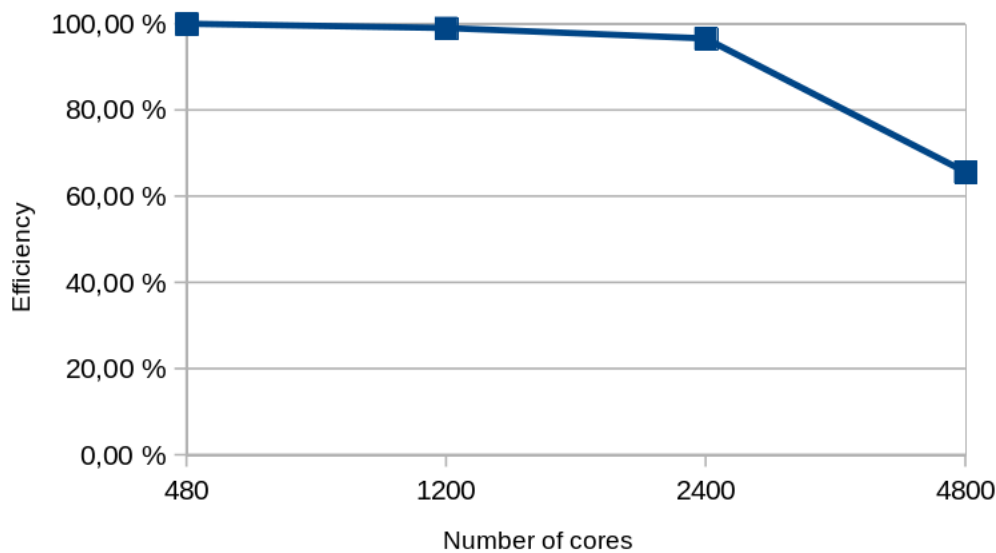


Figure 6: Strong scaling version 3.0 with resolution 16

## Conclusions

The version developed during the project has enabled the code to run on multiple nodes and being able to run simulations with higher resolutions, which were impossible to run on a reasonable time on the initial version of the code. Also, due to the flexibility of the code to run with multiple nodes, the code is capable of being used on a cloud infrastructure commonly used by the SME as on an HPC project.

From a light point of view, it has been shown that higher results of IES TM-30-18 Rf and M/P ratio can be obtained by increasing the resolution. In some cases, improvements of about 6% can be achieved by going from a resolution of 16 steps to a resolution of 32 steps. However, as the resolution increases, the system is closer to reaching the absolute minimum or maximum, so the improvement tends to be smaller for higher resolutions. These improvements are not only important in hospital environments, where patients are exposed 24 hours a day to artificial lighting, but can also be important in other areas such as museum lighting, where the chromatic reproduction of works of art is essential to see the colours in an unaltered way.

## References

- [1] G.C. Brainard et al. Action spectrum for melatonin regulation in humans: evidence for a novel circadian photoreceptor. *Journal of Neuroscience*. 2001; 21(16): 6405-6412. doi: 10.1523/JNEUROSCI.21-16-06405.2001
- [2] C. Blume et al. Effects of light on human circadian rhythms, sleep and mood. *Somnologie - Schlafforschung und Schlafmedizin*. 2019; 23(3): 147–156. doi: 10.1007/s11818-019-00215-x

## Acknowledgements

This work was financially supported by the PRACE project funded in part by the EU's Horizon 2020 Research and Innovation programme (2014-2020) under grant agreement 823767.