Strojniški vestnik - Journal of Mechanical Engineering 62(2016)5, 307-317
DOI:10.5545/sv-jme.2015.3234
Original Scientific Paper
Received for review: 2015-11-24
Received revised form: 2016-03-08
Accepted for publication: 2016-03-21

Using Newton's Method to Model the Spatial Light Distribution of an LED with Attached Secondary Optics

David Kaljun[1,*] – Jože Petrišič[1] – Janez Žerovnik[1,2]
[1]University of Ljubljana, Faculty of Mechanical Engineering, Slovenia
[2]Institute of Mathematics, Physics and Mechanics, Slovenia

In the design of optical systems based on light emitting diode (LED) technology, a crucial task is to handle the unstructured data describing the properties of optical elements in standard formats. This leads to the problem of data fitting within an appropriate model. Newton's method is used as an upgrade of the previously developed most promising discrete optimization heuristics showing an improvement in both performance and the quality of solutions. This experiment also indicates that a combination of an algorithm that finds promising initial solutions as a preprocessor and Newton's method may be a winning idea, at least on some datasets of instances.
Keywords: least squares function fitting, Newton's method, discrete optimization, local search, light distribution, LED

Highlights
- *Model for data fitting of LED photometry with the evaluation function is presented.*
- *The effects of a numerical method in conjunction with heuristics are studied.*
- *Algorithms are developed with the use of C++ programing language.*
- *The success of developed algorithms is tested on real and artificial datasets.*
- *The results are statistically evaluated.*
- *The numerical Newton's method prevails on both datasets, and provides substantial runtime shortening.*

0 INTRODUCTION

The light emitting diode (LED) industry has been evolving rapidly over the past several years. The fast pace of research and development in the field has had a number of impacts. One of the results is the massive use and implementation of LED elements in all kind of luminaires. While some of these luminaires are designed for ambient illumination, the majority are technical luminaires that have to conform not only to electrical and mechanical safety regulations but also to regulations that define and restrict the photometry of a certain luminaire. This means that the photometry of a luminaire has to be defined prior to production. In order to do that efficiently and with minimal errors the design engineer must virtually test the luminaires performance. Tools that can be used for this (OpticsWorks **[1]**, LigthTools **[2]**, TracePRO **[3]**) do exist and they offer a vast repository of sub-modules to develop and design custom lenses, reflectors, light guides, etc. However, these universal tools do not completely exploit the luminaire design possibilities that were introduced by the transition from conventional light source technologies to LED. One of the possibilities, which is also the main aim of a larger study that incorporates the research presented here, is to have an expert or intelligent system that would be capable of suggesting a secondary lens combination that would result in a user defined end photometry. In other words, the system would take some stock secondary LED lenses from different manufactures, place them on a defined LED array and search for the optimal combination of the lenses so that the resulting photometry would be as close as possible to the user defined one.

The method could enable the luminaire designer to custom design the light engine to a specific area of illumination, while keeping the mechanical and electrical parts of a luminaire untouched. This would in turn provide a customer with a tailored solution that would guarantee maximum efficiency, lower prices, less light pollution and the possibility of individualizing the illumination effect while maintaining a consistent visual appearance of the luminaries. There are several optimization tasks related to the development of the above idea.

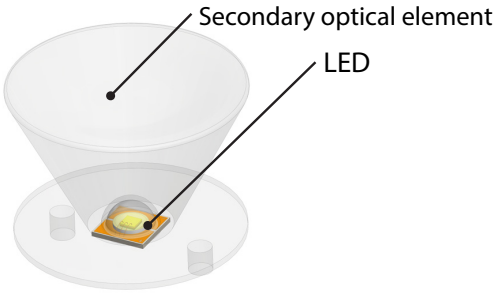

Fig. 1. *LED with attached secondary lens*

*Corr. Author's Address: University of Ljubljana, Faculty of Mechanical Engineering, Aškerčeva 6, 1000 Ljubljana, Slovenia, david.kaljun@fs.uni-lj.si

Here we focus on the approximation of spatial light distribution with a moderate number of suitable basis functions ([4] and [5]). The problem that is defined formally in the next section is motivated by the following. The data describing the properties of the lenses and/or of the desired light distribution is nowadays usually given in some standard format files that correspond to the measured (or desired) values at a number of points in space. This results in relatively large data files of unstructured data. Clearly, if the data can be well enough approximated i.e. as a linear combination of certain basis functions, this may enable faster computations using less computer storage. Indeed, for some special cases including LED lenses with symmetric light distribution, it is possible to find reasonably good approximations quickly (8 minutes' runtime on an Intel Core I7-4790K CPU @ 4 Ghz, the code is written in C++ and is not fully optimized). Sufficiently good approximation here means 2 % to 5 % *RMS* error (to be defined later) for target light distribution, taking into account expected noise in measurement using current technology. Recent experiments show that sufficiently good approximations can be obtained using some basic optimization algorithms, including local search algorithms and genetic algorithms ([6] to [8]). However, when using predefined lenses to design a luminaire that closely approximates a desired light distribution, it may be necessary for the approximation error to be much lower. The same task can also be seen as solving a problem of data compression, replacing a long unstructured data file with a much shorter one, in this case a sequence of parameters. It makes sense to aim at 0 % approximation when considering the data compression task. As the functions to be approximated are smooth, it is natural to try to improve the basic discrete optimization methods with continuous optimization techniques, i.e. Newton's method [9]. Here we consider Newton's method both as a standalone (restarted) algorithm and as a post-processor of other algorithms. The datasets used for testing and analysis are a selection of real lenses as used in previous studies and an artificial dataset that is large enough for statistical analysis. The artificial dataset is also generated in a way which assures that 0 % approximation is possible. Note that we have no guarantee that the realistic lenses can be approximated within our model with an arbitrary low *RMS* error. The rest of the paper is organized as follows: in Section Two we discuss the problem and present the mathematical model, Section Three is all about the algorithms and Newton's method implementation, Section Four presents the datasets used in the experiment, Section Five provides the experiment set-up, Section Six unveils the results, and Section Seven wraps everything up in the conclusion. The Appendix provides some formal details related to the application of Newton's method.

1 THE MODEL

The method mentioned above seems natural and straightforward, but looking closer, we observe some fundamental problems related to the realization of the main idea. Namely, both the spatial light distribution of LED lenses and the desired illumination are given in the standard data formats, which are just long unstructured lists of data. In particular, when the aim is to construct a lighting system that provides the desired illumination of the environment, it is necessary or at least very convenient to have the data in some more structured format. It is known that the spatial light distribution of some LED lenses can be approximated by the sum of a small number of certain basis functions [4]. Provided the approximation is sufficiently good, it may be possible to provide designs combining several lenses with a controlled error rate. This naturally opens several research avenues. For example, it is important to have error free or at least very good approximations of the basic lenses, and to have methods that are stable, in the sense that they are not too sensitive to the noise in the presentation of basic elements. Here we focus on the first abovementioned task, approximation of the unstructured spatial light distribution data. We search for an approximation of the Luminous intensity $I(\Phi, \boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})$ at the polar angle of Φ in the form:

$$I\left(\Phi, \boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}\right) = I_{max} \sum_{k=1}^{K} a_k cos^{c_k}\left(\Phi - b_k\right), \qquad (1)$$

where K is the number of functions to sum and a_k, b_k, c_k are the function coefficients that we search for. For brevity, coefficients are written as vectors $\boldsymbol{a}=(a_1, a_2, ..., a_K)$, $\boldsymbol{b}=(b_1, b_2, ..., b_K)$, $\boldsymbol{c}=(c_1, c_2, ..., c_K)$. The interval range of the coefficients is: $a=[0, 1]$, $b=[-90, 90]$, $c=[0, 100]$. Discrete optimization algorithms will work on the finite subsets where the possible values will be: $\boldsymbol{a}_* \in \{0, 0.001, 0.002, ..., 1\}$, $\boldsymbol{b}_* \in \{-90, 89.9, 89.8, ..., 90\}$, $\boldsymbol{c}_* \in \{0, 1, 2, ..., 100\}$.

Here we need to note two restrictions on the model. The first restriction emerges from the LEDs physical design. LED's cannot emit any light to the back side which is the upper hemisphere in our case. That is why all intermediate values that are calculated at the combined angle $(\Phi - b_k)$ greater than 90° equal 0. The second restriction deals with the

slightly unusual description of the light distribution in standard files such as Elumdat (file extension .ldt) [10] and IESNA (.ies) [11]. These files at present provide measured candela values per angle Φ on so called C planes which can be observed on Fig. 2. One C plane is actually only one half of the corresponding cross-section and does not describe the other half. But from a physical point of view we need to consider the impact from the other half of the cross-section.

Taking into account that all lenses used here are symmetric, we can simplify the calculation of the intermediate values and incorporate the impact of the other half by mirroring (multiplying by –1) all values that are calculated with the combined angle $(\Phi - b_k)$ less than 0°. Note however that this only works with symmetrical distributions, and should be reconsidered carefully when the method is to be applied to asymmetrical distributions.

The goodness of fit is defined as the root mean square error, formally defined by the expression:

$$RMS(\boldsymbol{a},\boldsymbol{b},\boldsymbol{c}) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left[I_m(\Phi_i) - I(\Phi_i,\boldsymbol{a},\boldsymbol{b},\boldsymbol{c})\right]^2}, \quad (2)$$

where N is the number of measured points in the input data, $I_m(\Phi_i)$ the measured Luminous intensity value at the polar angle Φ_i from the input data, and $I(\Phi_i, \boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})$ the calculated Luminous intensity value at the given polar angle Φ_i *RMS* represents the error of the approximation. Later in tables we provide the relative *RMS* error (*RMSp*) defined by Eq. (2a).

$$RMSp(\boldsymbol{a},\boldsymbol{b},\boldsymbol{c}) = \frac{100 \times N \times RMS(\boldsymbol{a},\boldsymbol{b},\boldsymbol{c})}{\sum_{i=1}^{N}\left[I_m(\Phi_i)\right]}[\%]. \quad (2a)$$

Remark. The model was successfully applied to LED's with attached secondary optics and symmetric light distribution [4] showing that sufficiently good approximations (*RMS* error below 5 %) can be obtained using a sum of only three functions, $K = 3$. Approximation of spatial light distribution of a LED with uniform distribution and without a secondary lens using this type of functions was first proposed in [5]. The model was slightly modified in [4] where a new normalizing parameter was introduced and, consequently, all other parameters will have values at fixed intervals known in advance. It should be noted that the modified model is equivalent to the original, only the number of parameters and their meaning differ. It is interesting to note that due to the symmetries in the examples, $K = 3$ is sufficient for both applications ([4] and [5]). In the general case, we expect that $K > 3$ functions will be needed for sufficiently good approximations, and in view of the optimization of the design of a luminaire it is interesting to have an idea how large the parameter K can grow to assure that the light distribution fits the desired (and/or standard) sufficiently well. We do not address this question here.

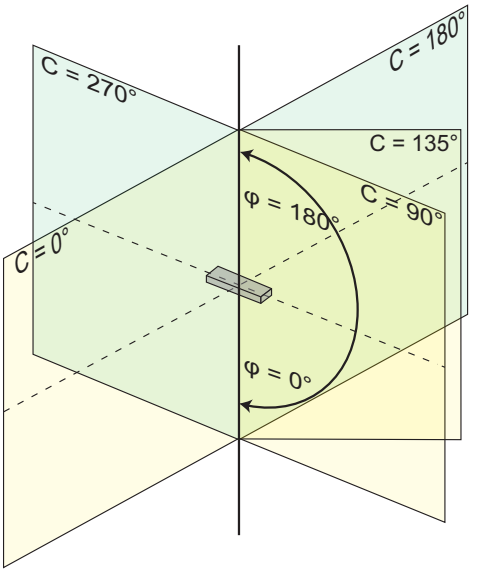

Fig. 2. *C-planes according to standard; C-planes angles: 0° to 360° | Φ angles: 0° to 180°*

When applying the model to the data compression problem, the target *RMS* error is 0 %. Therefore, we aim to improve the approximation results that were obtained previously ([6] and [7]) and restrict attention to the symmetric light distributions. We also fix $K = 3$ functions in the model. Besides the dataset of 14 realistic lenses that was used in some previous studies, here we also generate an artificial dataset in which a sample is simply a sum of three basis functions with randomly chosen parameters. This assures that a zero error approximation is possible for the instances of the artificial dataset. We are interested first in minimizing the approximation error and second in the computational time of the methods. In the next section we briefly outline the algorithms we use in the experiments.

2 THE ALGORITHMS

In previous work ([6] and [7]), the model described above was applied in conjunction with several custom built algorithms that are based on local search heuristics and some meta-heuristics. The algorithms implemented include a steepest descend algorithm, two iterative improvement algorithms with

different neighborhoods, and two genetic algorithms, a standard one and a hybrid one in which the best individuals of every generation are optimized with the iterative improvement algorithm. For a more detailed description of the algorithms we refer to ([6] and [7]). The results of the experiments showed that all of the algorithms applied are capable of providing satisfactory results on all tested instances, and differed mainly in computational time needed. The average *RMS* values obtained on real lenses were around $RMS = 2\,\%$. Hence, the results mentioned proved that the model is accurate and that sufficiently good approximations can be found with a variety of algorithms. However, recall that the model can also be used for data compression task. Zero or very low *RMS* error is also essential in the foreseen application, in which the pre-manufactured lenses are to be combined into a more complex luminaire with prescribed light distribution. In the model we use a sum of functions that are smooth and hence the first and second derivatives can be calculated allowing application of continuous optimization methods, in addition to the general discrete optimization meta-heuristics that were used before. We have chosen to use Newton's (also known as Newton - Raphson) iterative method [9] to find the solution that we seek. It is understood that the convergence of Newton's method largely depends on the initial solution. Therefore, we have applied the method in two ways. First, we use Newton's method as an optimizer which will pinpoint the local minimum of the solutions found by the heuristic algorithms. In a sense this implementation of Newton's method will be an extension of the discrete optimization algorithm, used to finalize the search to end in a local minimum. (Note that the local minima may be missed by the discrete optimization algorithms due to predefined length of the discrete moves.)

Secondly, we use Newton's method as a standalone algorithm that will on initialization generate a number of random (initial) solutions that are uniformly scattered over the whole search space and then the algorithm will use Newton's method on a number of best initial solutions to find the local minima. Of course, for both implementations to be comparable, the iteration count has to be controlled so that the overall maximum amount of computation time will be roughly the same.

Preprocessor multi-start IF. The **multi-start iterative improvement with fixed neighborhood (IF)** algorithm ([7] and [8]) first initializes several initial solutions. The initial solutions are randomly chosen from the whole search space. Each of the initial solutions is then optimized using the following steps. In the beginning the search step values (step for numerical differentiation) $da = 0.01$, $db = 1$ and $dc = I_{max}/10$, are initialized, giving 512 neighbors of the initial solution: ($a_1 \pm da$, $b_1 \pm db$, $c_1 \pm dc$, $a_2 \pm da$, $b_2 \pm db$, $c_2 \pm dc$, $a_3 \pm da$, $b_3 \pm db$, $c_3 \pm dc$).

Then the algorithm randomly chooses a neighbor, and immediately moves to the neighbor if its *RMS* value is better than the current *RMS* value. If no better neighbor is found after 1000 trials, it is assumed that no better neighbor exists. In this case the algorithm morphs the neighborhood by changing the step according to the formula $d_{i+1} = d_i + d_0$. More precisely, $da_{i+1} = da_i + da_0$ where da_0 is the initial step value. Analogously for *db* and *dc*. This is repeated until $i = 10$. If there still is no better solution, the initial step value is multiplied by 0.9 and the search resumes from the current solution with a finer initial step. The algorithm stops when the number of generated solutions reaches T_{max}.

Newton's method. Newton's method ([9], [12] and [13]) is a well-known numerical optimization method that can provide very good results under certain assumptions on the evaluation function and on the initial solution. Newton's method indirectly minimizes the evaluation function by looking for a solution for a system of nonlinear equations (first derivatives of the evaluation function). Newton's method solves the system of nonlinear equations iteratively by approximating it with a system of linear equations in each step which produce the delta vector. The delta vector is a part of the iterative scheme $x_k^{i+1} = x_k^i - d_k^i$. Newton's method converges when the delta vector vanishes, $d = 0$. At this point the evaluation coefficients found are the local minimum. Details are given in the Appendix. An obvious assumption is that the evaluation function has to be a continuous non-linear function for which the first and second order derivatives are defined. The initial solution has to be close enough to a local or global optimum for Newton's method to converge. Hence, the method may be very sensitive to the choice of the initial solution.

3 THE DATASETS

The experimental study uses two batches of instances, a dataset of 14 instances that correspond to real LED lenses and a dataset of artificial instances generated for purpose of this experiment. The artificial lenses are used to obtain more conclusive results on the statistical test, because a sample of 14 is rather small and may provide statistically insignificant results. The

real lenses on the other hand show that the algorithms are useful in real life scenarios.

Real lenses. We have chosen 14 different symmetrical lenses which are meant to be used with a CREE XT-E series LED, from one of the world's leading lens manufacturer LEDIL from Finland. We acquired the photometric data from LEDIL's on-line catalogue [14]. The data was provided in .ies format, which we then converted to a vector list that is more suitable to use in our algorithms. LEDIL measured the individual lenses with a 1° polar precision on four C panels. This means that from every .ies file we extracted 720 vectors. As the lenses are symmetric we only needed one C panel and, because we are only working on the lower half of the sphere (DLOR), we end up with 91 vectors (counting the 0° vector) on which we approximate the model.

Artificial lenses. In the dataset of 100 artificial examples, each element in the dataset was generated as follows.

A value from an interval was generated using uniform random distribution. (Intervals are $[0,1]$, $[0,90]$, or $[0,10]$, depending on the parameter. More precisely, the random generator chose one of the values from the finite sets:

$$a_1, a_2, a_3 \in \{0, 0.001, 0.002, ..., 0.999, 1\},$$
$$b_1, b_2, b_3 \in \{0, 0.01, 0.02, ..., 89.99, 90\},$$
$$c_1, c_2, c_3 \in \{0, 0.1, 0.2, ..., 9.9, 10\}.$$

Then the function values or candela values were computed for each polar angle $\Phi \in \{0, 1, 2, ..., 89\}$. The candela values for polar angles $\Phi \in \{90, 91, 92, ..., 180\}$ were set to 0. The data was then encoded into an .ies file structure, yielding a data file in the same format as the real lenses have. Note that the data generated assure that in each case zero *RMS* error approximation is possible within our model. Second, the dataset of 100 samples is sufficiently large for a meaningful statistical analysis of the experimental results.

4 THE EXPERIMENT SETUP

Before we go ahead and explain the experimental set-up, let us first remember the evaluation function that is the basis of Newton's method [9]. We already showed that the goodness of fit is measured using the *RMS* value that is calculated from Eq. (2). From this we can define the evaluation function as:

$$E(\boldsymbol{a},\boldsymbol{b},\boldsymbol{c}) = \frac{1}{N}\sum_{i=1}^{N}\left[I_{max}\left(G_1(\Phi_i)+G_2(\Phi_i)+G_3(\Phi_i)\right)-I_m(\Phi_i)\right]^2, \quad (3)$$

$$G_k(\Phi) = a_k cos^{c_k}(\Phi - b_k), \quad (4)$$

and

$$RMSp(\boldsymbol{a},\boldsymbol{b},\boldsymbol{c}) = \frac{100\times N\times\sqrt{E(\boldsymbol{a},\boldsymbol{b},\boldsymbol{c})}}{\sum_{i=1}^{N}\left[I_m(\Phi_i)\right]}[\%].$$

Here E represents the error to be minimized, N the number of measured points in the input data, I_{max} the maximum candela value, and $I_m(\Phi_i)$ the measured Luminous intensity value at the polar angle Φ_i from the input data. The experiment was set-up to provide data from different algorithms. This in turn enables an objective comparison and a statistical test to determine the best algorithm. Recall that we implemented Newton's method in two distinct ways. The first implementation uses the multi-start version of iterative improvement (**IF**) to find a good approximation which is then optimized via Newton's method. The second implementation uses the random generator to generate initial solutions of which 100 best are optimized with Newton's method. Table 1 shows different algorithms that were prepared for the experiment. After each run Newton's optimization method is applied.

Table 1. *Experiment algorithms*

	Config.	Algorithm	Multi-start	IF steps
Short runs 1 million	1	S-Newton	1000000	NA
	3	IF10	10	100000
	4	IF20	20	50000
	5	IF50	50	20000
	6	IF100	100	10000
Long runs 4 million	2	L-Newton	4000000	NA
	7	IF40	40	100000
	8	IF80	80	50000
	9	IF200	200	20000
	10	IF400	400	10000

Time. We ran the algorithms for two different lengths of time. The short run evaluates approximately one million possible solutions per instance (lens) in just under 45 s, and the long run approximately four million possible solutions per instance in about 3 minutes on a Core I7 - 4790K CPU. Newton's method took an average of 3 to 4 iterations to converge, which means that the time it took to run Newton's method is negligible in comparison to the time it took the whole algorithm run. Expressed in seconds, the Newton's method took approximately 2×10^{-3} s, opposed to minutes of CPU for the heuristics. In addition to the different time/iteration spans we ran the algorithms on two instance sets.

Datasets. Recall the two datasets of instances explained above, the dataset of 14 real lenses and the dataset of 100 randomly generated artificial instances.

Algorithms. We apply Newton's method both as a standalone algorithm (restarted on a selection of randomly generated initial solutions) and as a final step after discrete local search algorithm (**IF**) outlined above. There are several algorithms that vary in the number of multi-starts (or, equivalently in the length of each local search). Depending on the length (short run, long run) and the number of restarts we denote the algorithms **IF10, IF20, IF50, IF100** and **IF40, IF80, IF200, IF400**. The versions without a local search are denoted by **S-Newton** and **L-Newton** for short and long runs, respectively. See Table 1.

5 EXPERIMENTAL RESULTS

We begin the section with a comparison of the raw experimental data followed by the performance (quality of results) ranking and finish with the results of the Wilcoxon Signed rank test.

Experimental results are given in Figs. 3 to 6 and are summarized in Tables 2 and 3.

Comparison of the algorithms based on raw experimental results. Obviously, the pure multi-start Newton's method is by far the best in artificial instances. On the real lenses, the situation is a bit different. The Iterative improvement on several occasions outperforms Newton's method with random initial solutions. On both datasets, the long run yields

Table 2. *Artificial lenses statistical data in RMSp for short and long runs*

	Alg.	Mean	Std. dev.	Min.	Max.
Short runs 1 million	S-Newton	1.38E–04	8.51E–05	3.51E–05	3.91E–04
	IF10	2.34E+01	3.04E+01	5.81E–05	2.43E+02
	IF20	1.12E+01	1.64E+01	4.96E–05	1.33E+02
	IF50	6.33E+00	9.22E+00	4.13E–05	6.87E+01
	IF100	7.23E–01	2.71E+00	3.49E–05	1.91E+01
Long runs 4 million	L-Newton	1.38E–04	8.51E–05	3.51E–05	3.91E–04
	IF40	1.21E+01	2.54E+01	4.89E–05	2.43E+02
	IF80	1.75E+00	4.30E+00	3.51E–05	1.95E+01
	IF200	4.04E–01	1.96E+00	3.51E–05	1.42E+01
	IF400	6.17E–01	2.41E+00	3.49E–05	1.89E+01

Table 3. *Real lenses statistical data in RMSp for short and long runs*

	Alg.	Mean	Std. dev.	Min.	Max.
Short runs 1 million	S-Newton	5.39E+00	3.74E+00	1.79E+00	1.58E+01
	IF10	3.45E+00	1.52E+00	1.29E+00	5.79E+00
	IF20	8.95E+00	8.30E+00	1.60E+00	3.17E+01
	IF50	6.46E+00	4.78E+00	1.60E+00	2.09E+01
	IF100	4.91E+00	2.46E+00	1.60E+00	9.35E+00
Long runs 4 million	L-Newton	4.61E+00	2.65E+00	1.13E+00	1.03E+01
	IF40	3.20E+00	1.44E+00	1.19E+00	5.79E+00
	IF80	3.13E+00	1.75E+00	6.81E-01	6.33E+00
	IF200	3.56E+00	2.06E+00	1.07E+00	8.02E+00
	IF400	3.90E+00	1.85E+00	1.38E+00	7.56E+00

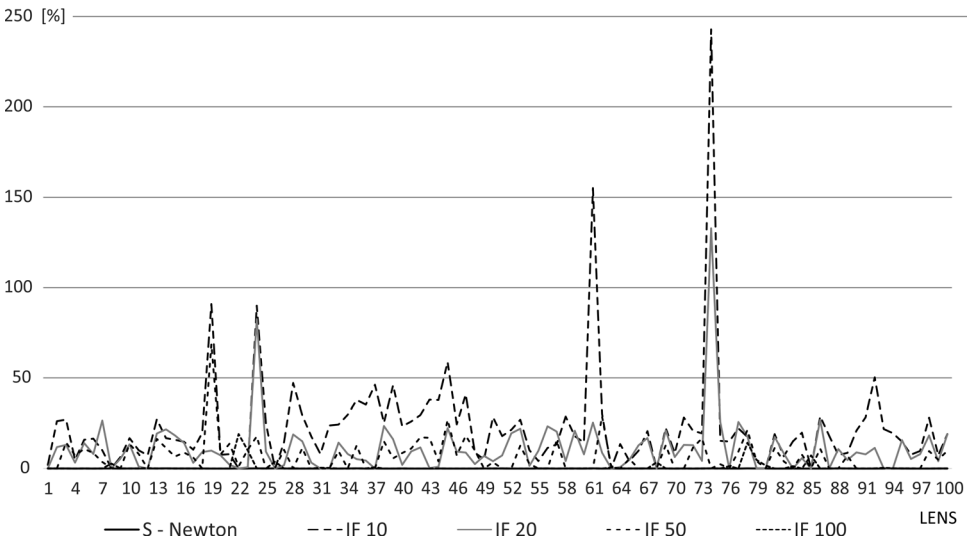

Fig. 3. *Best found solution on a short run; Artificial lenses per algorithm*

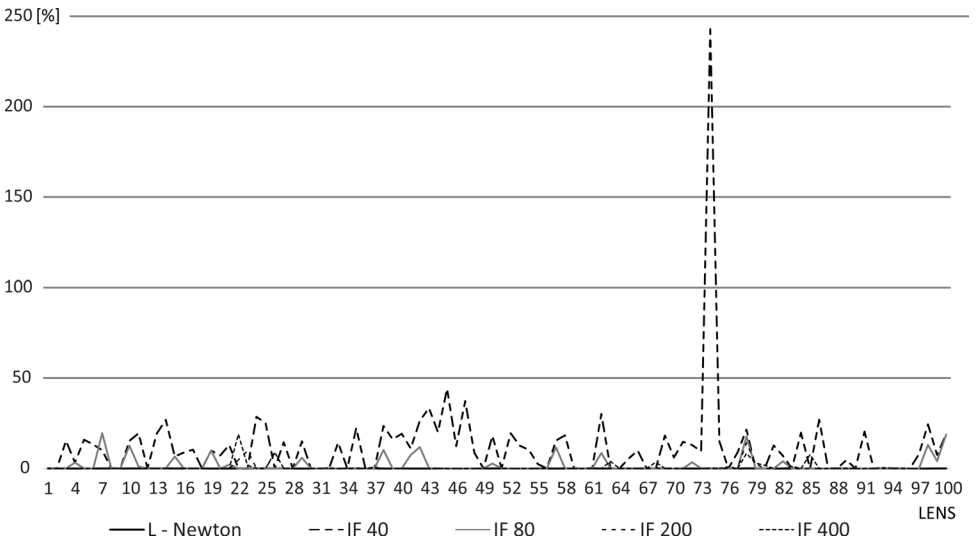

Fig 4. *Best found solution on a long run; Artificial lenses per algorithm*

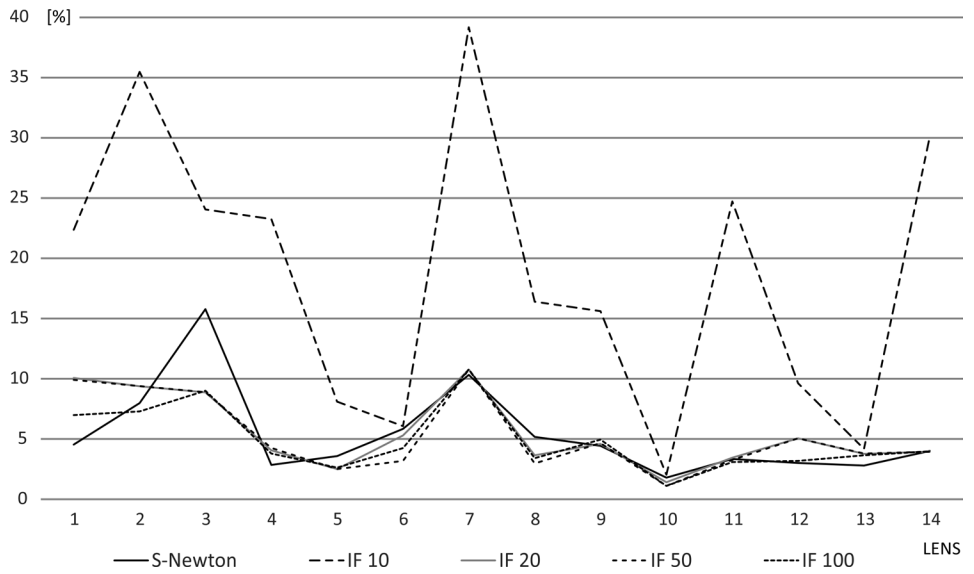

Fig. 5. *Best found solution on a short run; Real lenses per algorithm*

only slightly better results than the short run does (and the short run is executed four times faster). While we have no idea how far from optimal solutions the achieved values are for real lenses, we know that, by construction, a solution with 0 % *RMS* error exists for each of the artificial lenses. Because of that it is worth to note that on the artificial set, the random algorithms found nearly optimal solutions in all cases. The *RMS* errors are in the range of E-04, which still is not pure 0 % *RMS* error, but the very small difference could be due to rounding of the values in the .ies files. On the other hand, we did not find very low *RMS* values on the real set. The values that were found corresponded with the values of previous tests that were performed without any numerical assistance. We did however perform an experiment on the real set with a longer running time, in which we generated 16 million and 64 million initial solutions that showed similar behavior. The mean error and the minimum error over 14 lenses decreased under 3 % and 1.5 % with 16 million generated solutions, and under 2 % and 1 % *RMS* error after 64 million. Recall that 0 % error approximation may not be possible in these instances. While the success of Newton's method on artificial lenses is not surprising, it is not clear why the method is struggling on the realistic dataset. It may be that the

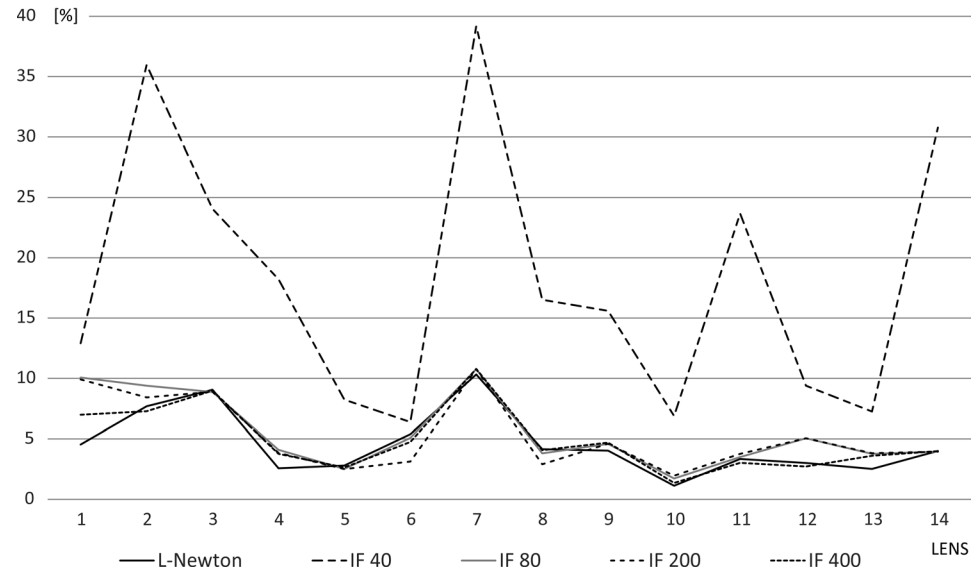

Fig. 6. *Best found solution on a long run; Real lenses per algorithm*

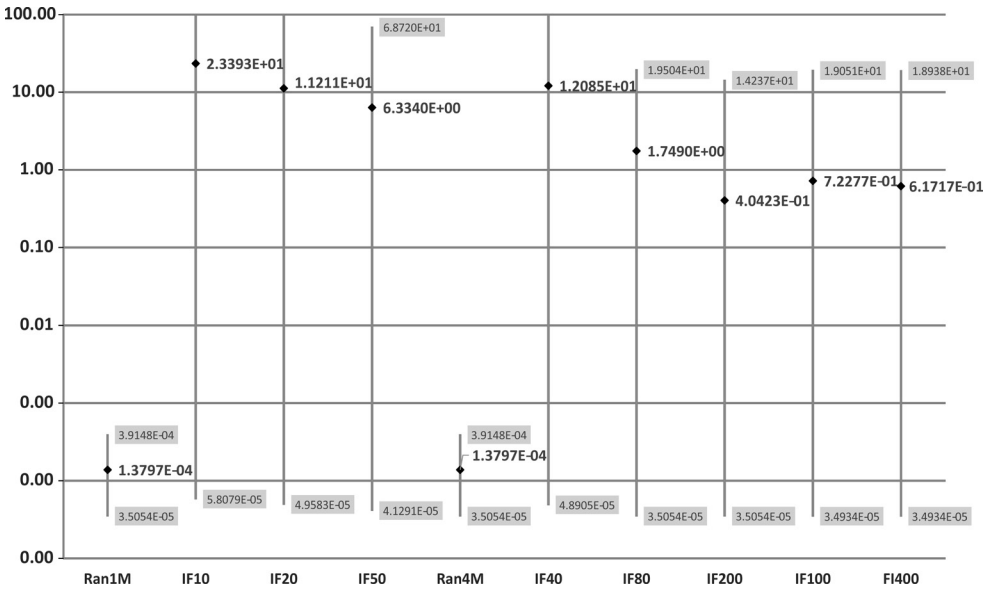

Fig. 7. *Min-Max scatter diagram for artificial lenses; (logarithmic y axis!)*

delta vector was too large due to a bad initial solution. One solution to that problem is the dampened Newton's method, which introduces a dampening factor to vector *d* to slow down the convergence and allow for more maneuvering space. But as we set out to evaluate the benefits of a standard Newton's method we did not implement any dampening in our algorithms.

The winners in this comparison are the same for both sets, but on the real set the differences between the random Newton and IF assisted Newton algorithms are a lot smaller than on the artificial set, we even see that on some instances the IF algorithms are better. This could be due to the fact that the IF algorithm was previously developed for real lenses that are from a limited range in search space and thus has a slight advantage on the set. The advantage of the pure Newton's method over the artificial dataset can be nicely observed from the data scatter in Fig. 7. We see that the IF algorithms provide a very high degree of data scatter whereas the random ones provide a very narrow result window. This may be due to the

Table 4. *Weighted ranking score of the algorithms*

RANK	Artificial				Real			
	Best		Mean		Best		Mean	
	Alg.	Score	Alg.	Score	Alg.	Score	Alg.	Score
1	L-Newton	944	L-Newton	1000	L-Newton	102	L-Newton	140
2	S-Newton	940	S-Newton	900	IF 100	97	S-Newton	124
3	IF 200	836	IF 20	588	IF 400	96	IF 10	114
4	IF 80	810	IF 100	585	IF 50	94	IF 50	89
5	IF 100	678	IF 10	559	S-Newton	90	IF 20	77
6	IF 400	660	IF 50	538	IF 20	88	IF 100	72
7	IF 50	529	IF 400	385	IF 200	86	IF 40	61
8	IF 40	514	IF 40	358	IF 80	83	IF 80	39
9	IF 20	379	IF 80	304	IF 10	23	IF 400	34
10	IF 10	212	IF 200	283	IF 40	21	IF 200	20

nature of the search because the IF algorithms focus in one defined direction which may not be the best one. Because of that, Newton's optimization cannot escape the potential pitfall of the direction. In contrast, the randomly generated solutions generally find lower quality results, while at the same time providing more maneuvering space for Newton's method to find the best direction on more takes.

A comparison of the algorithms based on weighted ranking. We assign a weight from 1 to 10 to each instance solution per algorithm. If the algorithm found the best solution on an instance it would get the weight 10 and if it found the second best solution it would get the weight 9, and so on until 1 for the worst solution. The total score of the algorithm is the sum of the scores on each instance.

In the same way, we compute the score based on the average values per algorithm and lens. The results are presented in Table 4. Note that the ranking here compares both short and long runs. As expected, Table 4 confirms the superiority of the pure Newton's method with the artificial dataset. However, the situation is much more complicated on the dataset of real lenses. Despite Newton's methods (long and short run) being the two best when considering the average results, they are not both when looking at the best solutions! The long run Newton is still the overall best, but the short run Newton is in fifth place, outrun by two short IF algorithms and one long IF algorithm. We can also observe that the score differences are much smaller on the real set, which indicates that the pure Newton's method is not as superior as it was on the artificial set. The lesser superiority could be explained in part by the fact that the IF algorithms were developed using the real lens set. Hence the IF could have some unexpected advantages. However, Newton's method improves the results in all cases.

Wilcoxon test. The third comparison is based on the statistical paired signed Wilcoxon **[15]** rank test. This statistical test compares algorithms pair by pair to estimate the difference between them. This is done via the asymptotic difference. If the value of the asymptotic difference is lower than 0.05 then the algorithms in the pair significantly differ one from another. The asymptotic differences in the algorithm pairs is presented in Tables 5 to 8.

Table 5. *Asymptotic significances of Wilcoxon Signed rank test for results for short runs on artificial lenses*

algorithm	IF10	IF20	IF50	IF100	S-Newton
IF10		4.078E–11	1.020E–13	1.000E–13	1.000E–13
IF20			7.162E–04	3.120E–13	1.000E–13
IF50				3.628E–06	1.650E–13
IF100					3.234E–08

Table 6. *Asymptotic significances of Wilcoxon Signed rank test for results for long runs on artificial lenses*

algorithm	IF40	IF80	IF200	IF400	L-Newton
IF40		1.736E–11	3.917E–11	7.950E–09	1.899E–12
IF80			3.269E–02	5.563E–01	3.411E–06
IF200				4.818E–06	7.044E–05
IF400					1.176E–08

Table 7. *Asymptotic significances of Wilcoxon Signed rank test for results for short runs on real lenses*

algorithm	IF10	IF20	IF50	IF100	S-Newton
IF10		9.815E–04	9.815E–04	9.815E–04	9.815E–04
IF20			4.326E–01	4.133E–02	5.936E–01
IF50				5.098E–01	9.750E–01
IF100					4.703E–01

Table 8. Asymptotic significances of Wilcoxon Signed rank test for results for long runs on real lenses

algorithm	IF40	IF80	IF200	IF400	L-Newton
IF40		9.815E–04	9.815E–04	9.815E–04	9.815E–04
IF80			5.509E–01	5.553E–02	3.546E–02
IF200				4.703E–01	1.240E–01
IF400					4.703E–01

A look over the Wilcoxon test results reveals that there are mostly no similarities between the algorithms when they ran on the artificial set. We can see that the asymptotic significance values are very low, which means that there are significant differences between algorithms in pairs. We do however have one exception in the pair IF 80 to IF 400, where the asymptotic difference is just over the margin, so we could say that these two have some similarities. The story is completely different on the real dataset, where we can find that most IF algorithms are similar to random algorithms. Thus, based on the statistical test we cannot conclude that either of them is superior. This also corresponds with the findings of the ranking and *RMS* error comparison. The Wilcoxon test provided similar conclusions as the previous tests did, but we need to be careful because the data sets differ in size and the real lenses set can be a bit inconclusive as it is a rather small sample with only 14 instances. That is why the artificial lenses with 100 instances could give a more accurate result.

Table 9. *Real lens RMSp for RAN 4M with and without Newton's method; Quality increase* Δ

Instance	RAN 4M	Newton	Δ [%]
CP12632	27.996	7.6908	72.53
CP12634	45.8986	9.05513	80.27
CP12633	10.7706	2.57185	76.12
CA11934	10.2492	2.7982	72.70
CA11268	15.1818	5.38851	64.51
CP12817	29.6252	10.3279	65.14
CA11265	9.6437	4.1553	56.91
CP12636	7.62895	4.03813	47.07
CA13013	2.70647	1.12548	58.42
FP13030	10.1866	3.34882	67.13
CA11525	12.0224	3.00557	75.00
CA12392	6.87747	2.51916	63.37
CA11483	24.5813	4.0032	83.71

6 CONCLUSION

Here we presented an upgrade of a previously developed most promising discrete optimization heuristics with a continuous optimization method. It was shown that the application of Newton's method led to an improvement of both performance and quality of solutions. In terms of raw performance, we got from the initial 8 minutes' runtime for one algorithm on one lens to an approx. 45 s runtime using the upgraded IF 10 or S-Newton algorithm. The stated runtime is accurate for symmetric lenses and an input of 91 vectors. When working on asymmetric lenses the input will be around 33,000 vectors, and this is when the problem becomes a big data problem. Because of the algorithms' design, the runtime is expected to increase to about 2 hours and 15 minutes. The increase will be by a factor of 180, while the number of vectors is increased by a factor 360. On the asymmetric lenses, the runtime will be lowered from around 24 hours to 2 hours and 15 minutes. Despite the drastic time shortening the quality of the solutions was not worse thanks to Newton's method, which enabled us to find local minimums on the majority of solutions found by the heuristic algorithms. In fact, Newton's method successfully minimized the *RMS* error on all of the experiment cases with the average of 60 % increased quality (minimized *RMS*) over previous experiments done in ([4] and [7]). This can be well observed in Table 9, where we can see the *RMS* error found by the RAN 4M algorithm before the application of Newton's method and after. We can conclude that the integration of a numerical approach with previously developed heuristics significantly improved the application performance to the level at which it is useful in the main research. On the other hand, we have learned that due to the sensitivity of Newton's method to the choice of initial solutions, it may be rewarding to use a preprocessor that may provide promising initial solutions. In particular, on the dataset consisting of real lenses, the experiment showed that the initial solutions provided by a discrete local search algorithm improved the overall performance of the algorithm. This leads to the conclusion that a combination of an algorithm that finds promising initial solutions as a preprocessor to Newton's method may be a winning combination, at least on some datasets of instances. Hence, in a practical application, it may be worth developing good heuristics that may handle specific properties of the instances and thus provide promising initial solutions for final optimization.

7 ACKNOWLEDGEMENT

This work was supported in part by ARRS, the Research agency of Slovenia, grants P1-0285 and ARRS-1000-15-0510. We sincerely thank two

anonymous reviewers for detailed reading of the manuscript and for constructive remarks.

8 REFERENCES

[1] Optis, Optisworks, from *http://bit.ly/1CjKp4t*, accessed: on 2015-06-07.

[2] Lighttools, from *http://optics.synopsys.com/lighttools/*, accessed on 2015-06-07.

[3] Lambardes, Tracepro, from *http://www.lambdares.com/*, accessed on 2015-06-07.

[4] Kaljun, D., Žerovnik, J. (2014). Function fitting the symmetric radiation pattern of a led with attached secondary optic. *Optics Express*, vol. 22, no. 24, p. 29587-29593, **DOI:10.1364/OE.22.029587**.

[5] Moreno, I., Sun, C.-C. (2008). Modeling the radiation pattern of leds. *Optics Express*, vol. 16, no. 3, p. 1808-1819, **DOI:10.1364/OE.16.001808**.

[6] Kaljun, D., Žerovnik, J. (2014). On local search based heuristics for optimization problems. *Croatian Operational Research Review*, vol. 5, no. 2, p. 317-327, **DOI:10.17535/crorr.2014.0016**.

[7] Kaljun, D., Poklukar, D.R., Žerovnik, J. (2015). Heuristics for optimization of led spatial light distribution model. *Informatica*, vol. 39, no. 2, p. 317-327.

[8] Kaljun, D., Žerovnik, J. (2015). Developing led illumination optics design. Papa, G. (ed.), *Advances in Evolutionary Algorithms Research*. Nova Science Publishers, Inc., Hauppauge, p. 11788-3619.

[9] Quarteroni, A., Sacco, R., Saleri, F. (2015). Nonlinear systems and numerical optimization. Marsden, J., Sirovich, L., Antman, S. (eds.), *Numerical Mathematics*. Springer-Verlag GmbH, Heidelberg.

[10] EN 13032-1:2004+a1:2012. *Light and lighting - measurement and presentation of photometric data of lamps and luminaries - part 1: Measurement and file format*. CEN, Brussels.

[11] IESNA, LM-63-02. *Standard file format for the electronic transfer of photometric data and related information*, ANSI, Washington D.C.

[12] Thukral, R. (2008). Introduction to a Newton-type method for solving nonlinear equations. *Applied Mathematical Computation*, vol. 195, p. 663-668, **DOI:10.1016/j.amc.2007.05.013**.

[13] Herceg, D. (2013). Means based modifications of Newton's method for solving nonlinear equations. *Applied Mathematical Computation*, vol. p. 216, 6126-6133.

[14] Ledil oy. (2015). from *http://www.ledil.com/products/?y*, accessed on 2015-06-07.

[15] Wilcoxon F. (1945). Individual comparisons by ranking methods. *Biometrics*, vol. 1, p. 80-83, **DOI:10.2307/3001968**.

[16] Kaljun, D., Petrišič, J., Žerovnik, J. (2016). *Using Newton's method to model a spatial light distribution of a LED with attached secondary optics*, ArXiv:1603.01090.

9 APPENDIX

Jacobian matrix:

$$J(a_1,...,c_3)=\begin{bmatrix} \frac{\partial^2 E(a,b,c)}{\partial a_1 \partial a_1} & \cdots & \frac{\partial^2 E(a,b,c)}{\partial c_3 \partial a_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 E(a,b,c)}{\partial a_1 \partial c_3} & \cdots & \frac{\partial^2 E(a,b,c)}{\partial c_3 \partial c_3} \end{bmatrix}.$$

As the evaluation function is clear enough that it is not difficult to find the first and second derivatives using any of several available systems for symbolic computations, we only present the basic components of Newton's method here. An earlier version of the manuscript with an extended appendix is available at ArXiv **[16]**.

Delta vector:

$$d=\begin{bmatrix} da_1 & da_2 & da_3 & db_1 & db_2 & db_3 & dc_1 & dc_2 & dc_3 \end{bmatrix}.$$

Right side:

$$R(a_1,...,c_3)=\begin{bmatrix} \frac{\partial E(a,b,c)}{\partial a_1} & \cdots & \frac{\partial E(a,b,c)}{\partial c_3} \end{bmatrix}^T.$$

System of equations to solve for d:

$$J(x_i)\times d_i = R(x_i).$$

Coeficient vector:

$$x=\begin{bmatrix} a_1 & a_2 & a_3 & b_1 & b_2 & b_3 & c_1 & c_2 & c_3 \end{bmatrix}.$$

Iterative scheme:

$$x_{i+1}=x_i-d_i.$$