Automated Mode-Matching of Gaussian Beams

Matthew Argao

In the field of optics and laser related physics, Gaussian beams are a well documented model of electromagnetic radiation whose characteristics of transverse electric field and intensity are approximated by Gaussian functions of the form:

$$E(r,z) = E_0 \frac{w_0}{w(z)} \exp\left(\frac{-r^2}{w^2(z)} - ikz - ik\frac{r^2}{2R(z)} + i\zeta(z) \right) \ ,$$

Where r is the radial distance from the center axis of the beam, z is the axial distance from the beam's narrowest point, also termed waist, k is the wave number in radians per meter, E_0 is the electric field at E(0,0), w(z) is the radius where the field amplitude and intensity drop to 1/e and $1/e^2$ of their axial values respectively, w_0 is the waist size w(0), R(z) is the radius of curvature, and ζ (z) is the Gouy phase shift which is an extra contribution to phase seen in Gaussian beams. w(z) and R(z) can be further characterized by following equations:

$$w(z) = w_0 \sqrt{\left(1 + \left(\frac{z}{z_R}\right)^2\right)}$$

$$R(z) = z[1 + \left(\frac{z_R}{z}\right)^2]$$

Where z_R is the Rayleigh Length, defined as the distance along the propagation direction from w_0 where the area of the beam cross section is doubled. As many lasers emit beams that follow this Gaussian profile, it follows then that the beams encountered in much laboratory research are characterized by Gaussian beams. It becomes natural, then, for researchers to wish to output a desired Gaussian beam that meets certain specifications, determined on a case by case basis of the experiment. For example, when one wishes to inject a given pump laser beam into another laser cavity, it is required that the beam corresponds to the previous beam. In other words, one must mode-match the two beams. The simplest way in which we manipulate Gaussian beams is through the use of lenses; lenses transform an input Gaussian beam with certain parameters into another Gaussian beam which consists of different parameters. For our research, the main parameters of Gaussian beams which are focused on are the complex beam parameter, q(z), the beam's waist, w(z), and its radius of curvature, R(z). It becomes advantageous to represent the complex beam parameter in terms of its reciprocal to show the relationship between parameters q(z), w(z) and R(z), given as:

$$\frac{1}{q(z)} = \frac{1}{R(z)} - \frac{i\lambda}{\pi w^2(z)}$$

Where λ is the wavelength of the beam. These quantities are more easily visualized in Figure 1.

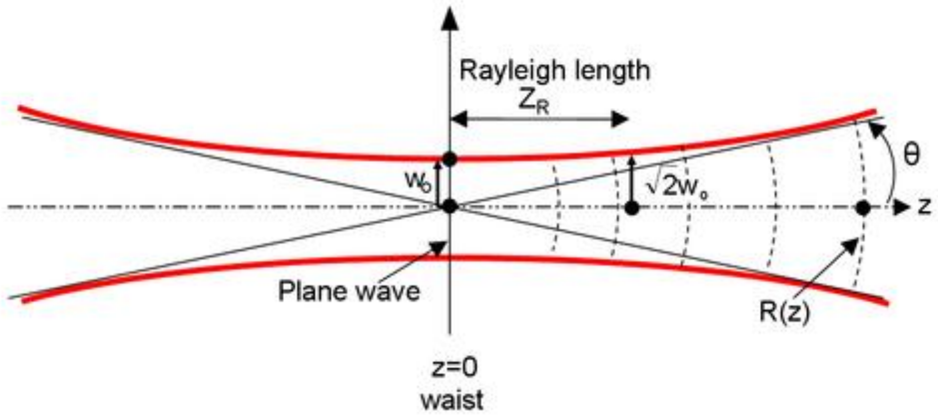

Figure 1. General profile of a Gaussian beam.

The complex beam parameter readily contains information on both the radius of curvature and waist of a Gaussian beam at any position of its propagation. Thus, the complex beam parameter becomes essential to the simulation of Gaussian beam propagation.

In ray transfer matrix analysis, we use the initial beam parameter, q_i, and the ABCD transfer matrix of the optical system to find the output beam, q_f. The ABCD transfer matrix is a matrix which characterizes optical elements, such as lenses or free space, with a matrix of the form:

$$\begin{pmatrix} A & B \\ C & D \end{pmatrix}$$

When determining the optical elements' effects on, say a laser, we consider two reference planes: the input and output planes. Using this expression:

$$\begin{pmatrix} x_2 \\ \theta_2 \end{pmatrix} = \begin{pmatrix} A & B \\ C & D \end{pmatrix} \begin{pmatrix} x_1 \\ \theta_1 \end{pmatrix}$$

We consider the ray, or beam, entering the input plane a distance x_1 from the optical axis (which is taken to coincide with the z-axis) at an angle of θ_1. As the beam travels it eventually crosses the output plane this time with characteristics x_2 and θ_2. An example of this can be seen through the propagation of a beam through free space. The ray transfer matrix of free space is given by:

$$S = \begin{pmatrix} 1 & d \\ 0 & 1 \end{pmatrix}$$

Where d is the distance traveled along the optical axis. Using the expression given previously, we get:

$$\begin{pmatrix} x_2 \\ \theta_2 \end{pmatrix} = \begin{pmatrix} 1 & d \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ \theta_1 \end{pmatrix}$$

Which we can then use to relate the parameters of the input and output rays visualized in Figure 2:

$$x_2 = x_1 + d\theta_1$$
$$\theta_2 = \theta_1$$

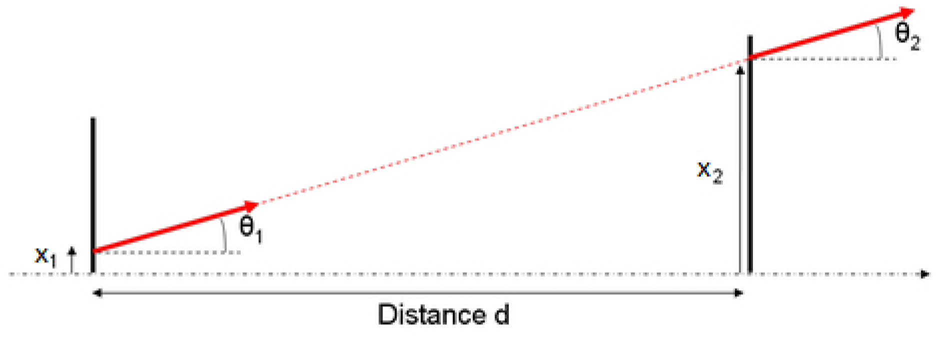

Figure 2. Free-space propagation example.

Similarly we can go through the same geometric model using the ABCD matrix of a thin-lens given as:

$$L = \begin{pmatrix} 1 & 0 \\ -\frac{1}{f} & 1 \end{pmatrix}$$

Where f is the focal length of the lens with positive values corresponding to convex or converging lenses. Similar to the geometric model we can apply this useful matrix formalism and apply it to describe Gaussian beams. Using the previously defined parameter q we apply the following equation:

$$\begin{pmatrix} q_2 \\ 1 \end{pmatrix} = k \begin{pmatrix} A & B \\ C & D \end{pmatrix} \begin{pmatrix} q_1 \\ 1 \end{pmatrix}$$

Where q_1 and q_2 are the input and output complex beam parameters respectively and k is the normalization constant chosen to ensure that the second component of the ray vector equals one. Upon expansion of the equation we receive:

$$q_2 = k(Aq_1 + B)$$

$$1 = k(Cq_1 + D)$$

Dividing the first equation by the second to eliminate the normalization constant and taking the reciprocal we finally get the general form:

$$\frac{1}{q_2} = \frac{C + \frac{D}{q_1}}{A + \frac{B}{q_1}}$$

The utility of the ABCD matrix formalism allows us to create optical systems consisting of multiple elements and lets us receive an output beam for every input beam. Due to the need to propagate Gaussian beams through these multiple matrix elements we chose MATLAB (Matrix Laboratory) to handle all propagation and mode-matching code.

Prior to my working on the program, Professor Eugeniy Mikhailov had compiled a program which simulated Gaussian beam propagation through free space and select lenses with chosen focal lengths. The program requires input of initial and final values of radius, waist and position. Combining these input parameters and specified positions and focal lengths for three lenses, the beam is profiled and visualized. Qualitative analysis of the plot allows judgment of whether a beam is mode-matched or not; forward and backward propagations of the beam with the same parameters and optical system should ideally yield identical beams. An example of a poorly mode-matched beam is shown in Figure 3.

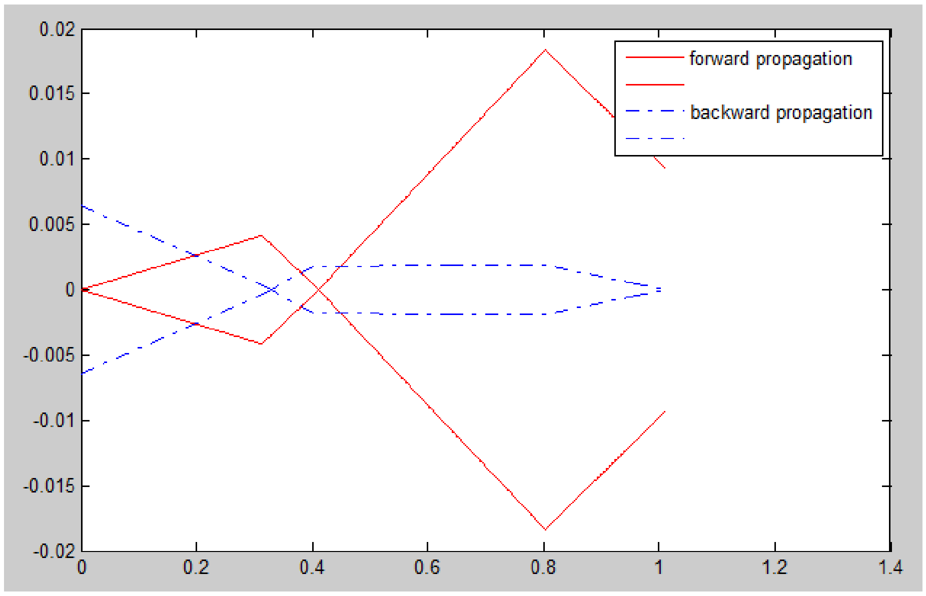

Figure 3. A poorly mode-matched beam solution. As seen qualitatively, the forward and backward propagation of the beam do not coincide at all. The x-axis is the propagation direction of the beam and the y-axis is the waist distance from the optical axis.

Keeping this in mind, we began to build upon the old code and automated the process by which mode-matching occurs. In the original program, the solutions for the placement of the focal lenses were already given thus to automate placement of the lenses we required a fitness function to guide the program into picking a solution which minimized "energy". The fitness function works on the basis of penalties; the further one deviates from the desired laser beam the higher the penalty, ergo higher penalty energies correspond to worse solutions. Four factors were taken into account, when determining the fitness of a solution. The solution must make sure the lenses are between the edge boundaries, make sure that lenses are not placed too close to each other, make a collimated region between the second and third lens and ensure that at a given point the forward and backward propagation of the beam have matching waists. The first and second factors of this list were needed to ensure that solutions given were possible to recreate in real life, the third factor was deemed necessary due to the practicality of having a collimated section in an optical system and the fourth factor is that factor that majorly contributes to the determination of whether beams were actually mode-matched. Figure 4 shows an example of what the fitness function looks like for a three lens solution. In this particular example, lens two is held at .403m and lens three is held at .803m and lens one is shuffled from 0 to x_f = 1.0107m.

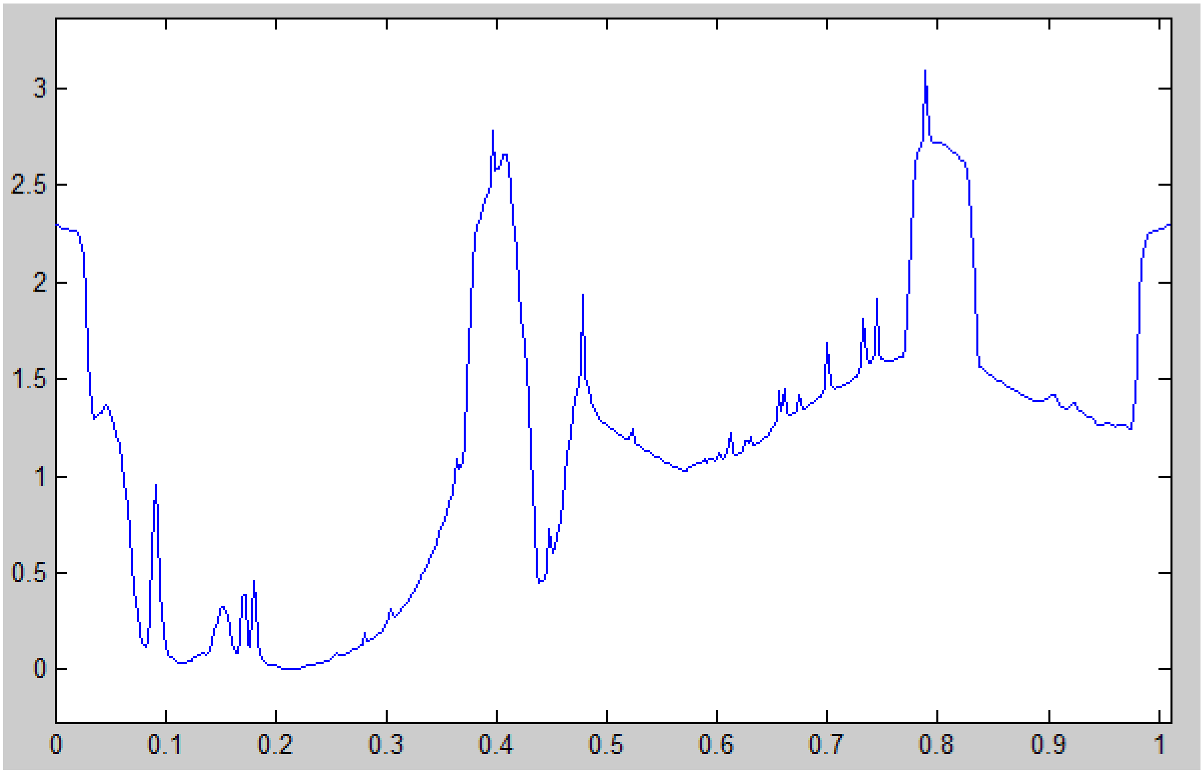

Figure 4. Fitness function for optical system where x_lens_two = .403m, x_lens_three = .803m. The x-axis is position of the first lens and the y-axis depicts the penalty attributed to that position. The effects of the various penalties can be seen within the plot; penalties associated with lens overlap are apparent in peaks ~.4m and ~.8m, and penalties associated with being within x_0 and x_f are seen in the peaks near the ends of the plot.

After properly instituting a fitness function, we iterated the lenses, taking care to preserve their order in the optical system, through different positions between the initial point, x_0, and the final point, x_f. For the example solution given in the original code, the order of the lenses is already given, however, in real life laboratory situations, researchers are usually dealing with a set of lenses, of which they choose to use. Thus it was necessary to implement a feature that permuted through all possible lens choices in a given “box” of lenses and iterated the same fitness through each permuted set. Going further along the direction of practicality, more customizable features were added to the code. Not only did it automatically pick and permute all possible combinations of every lens in the “lens set”, it also allowed settings for things such as number of shuffles to each set per iteration, how many unique solutions to display and threshold at which they’re determined to be “unique and separate”, and how many unique solutions to visualize. With all this to iterate through, efficiency of the code becomes a problem. So, again, we took this into consideration for usability and ensured that the code was optimized for multiple core processors which speed the runtimes significantly.

As of now, the automated program appropriately finds the correct solution that coincides with the given test solution of the original program. As an example of its functionality, we insert a given set of lenses consisting of .075m and .203m focal length lenses. After running, the program displays five solutions, coinciding with the various permutations of these two types of lenses. The first three solutions, coinciding with the lowest three energy solutions are as follows:

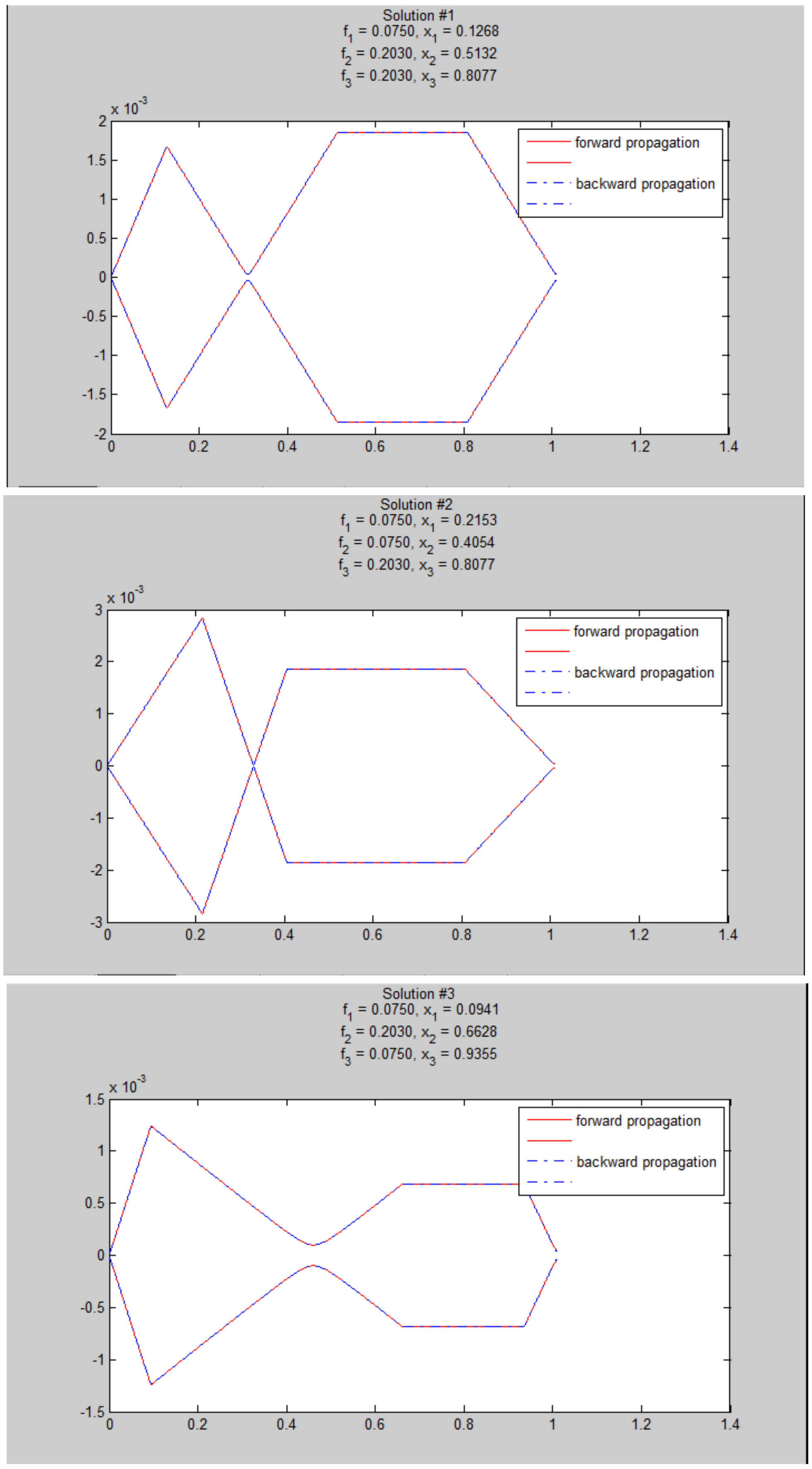

Figure 5. Three solutions found by the automated mode-matching program. The x-axis is the propagation direction of the beam and the y-axis is the waist distance from the optical axis.

The above example worked with a lens set that consisted of only two types of lenses. The number of permutations and possible combinations expand significantly with each new type of lens added to the set; what was once a 45-second calculation could easily become a multiple hour affair. Thus efficiency remains a primary concern for further development of this project. Revising the original code for beam propagation to work more efficiently will help alleviate the problems concerning runtime. The previous beam propagation code iterated through each test point individually which was not as efficient as it could be given what MATLAB could provide. Thus we sped up the propagation code by allowing chunks of elements to be computed in stages, first points to the left of the lens then after. This minor change decreased our runtimes by a factor of approximately 4, from a ~40s calculation to ~9s calculation. Future directions may include refinement of fitness function and additional features that would be consistent with making the program interface more intuitive for researchers who might use it. In the end, practicality, efficiency and accuracy are what we hope to achieve in the final product.

References

Alda, Javier. "Laser and Gaussian Beam Propagation and Transformation." Encyclopedia of Optical Engineering (2003): 999-1013.

Self, Sidney A. "Focusing of Spherical Gaussian Beams." APPLIED OPTICS 22.5 (1983): 658-61.