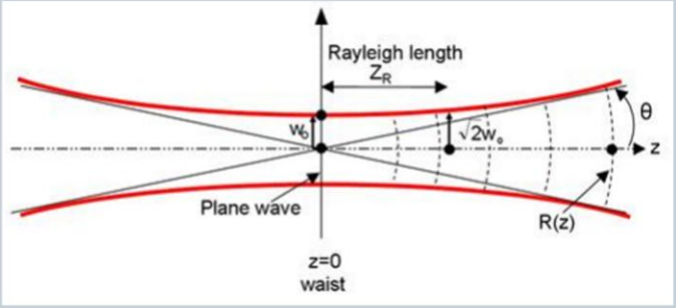
Automated Mode-Matching of Gaussian Beams

MATTHEW ARGAO

Introduction – Gaussian Beams

- Main Parameters:
  - w(z) – Waist
  - R(z) – Radius
  - q(z) – Complex Beam Parameter

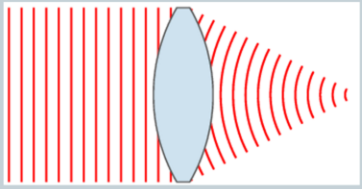


In the field of optics and laser related physics, Gaussian beams are a well documented model of electromagnetic radiation whose characteristics such as transverse electric field and intensity are approximated by Gaussian functions. Stemming from this model one can visualize the general shape of a Gaussian beam, shown here with a hyperbolic profile. Out of this we define some characteristics that differentiate one Gaussian beam from another. In particular we look at the waist, the radius and complex beam parameter. The radius is simply the Radius of Curvature of the wavefronts of the beam and the waist is a parameter which defines width of the beam.

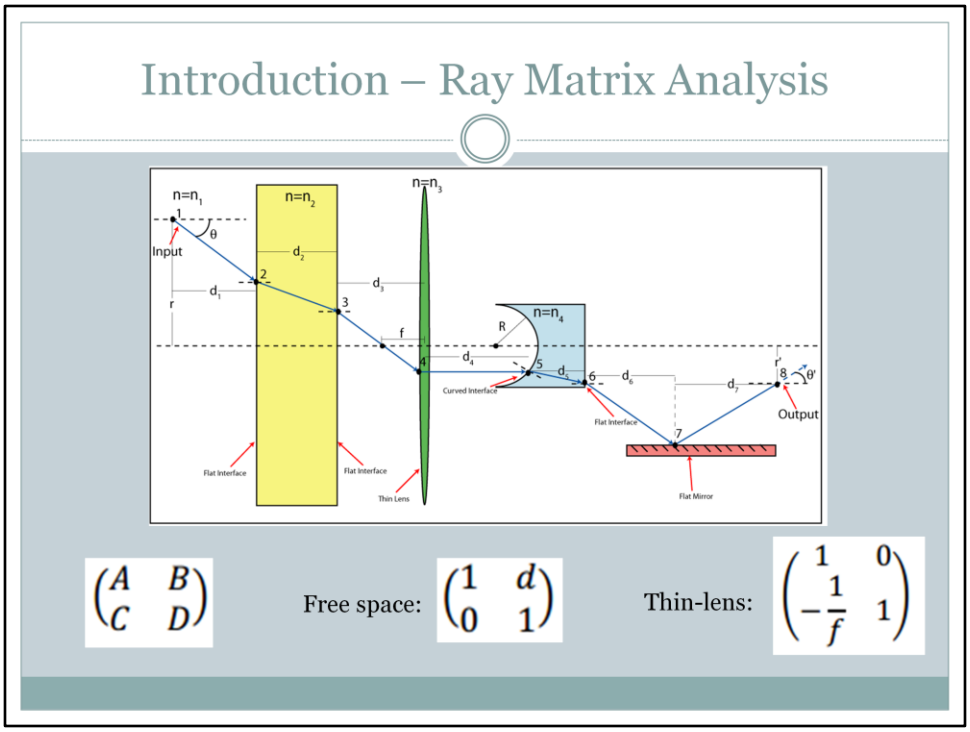
Introduction – Complex Beam Parameter

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

- How do we manipulate Gaussian beams then?



The complex beam parameter, however, is distinct from the other parameters in that it is characterized by the waist and radius of the beam. The relationship, shown above as a reciprocal, is invaluable to the analysis of Gaussian beams. We can take an input q parameter and using ray transfer analysis, profile the rest of the beam. Once we have the rest of the complex beam parameters we can easily extrapolate the waist and radius of the beam at that particular point in space. Which naturally leads us to want to manipulate Gaussian beams to output particular values for waist and radius characterized by q according to whatever an experiment requires. The most simple way to do this is through the use of lenses. As you can see in this small animation, the input beam has a given waist that seems to be uniform, consistent with the definition of a collimated beam. As such, its radius of curvature is seen to be almost flat, thus having a value which approaches infinity. However, once this input beam meets the lens it acquires a phase which causes the beam to become convergent.



In ray matrix analysis the various optical elements depicted in this diagram can be described by an ABCD matrix, shown below. The two types of optical elements we were interested in in this experiment were free space and thin lenses. However, I'd like to stress how general this method is through this diagram depicting other elements such as mirrors and other materials with a different index of refraction.

Introduction – Ray Matrix Analysis

- Ray Matrix Analysis:

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

- Components:

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

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

- General Form:

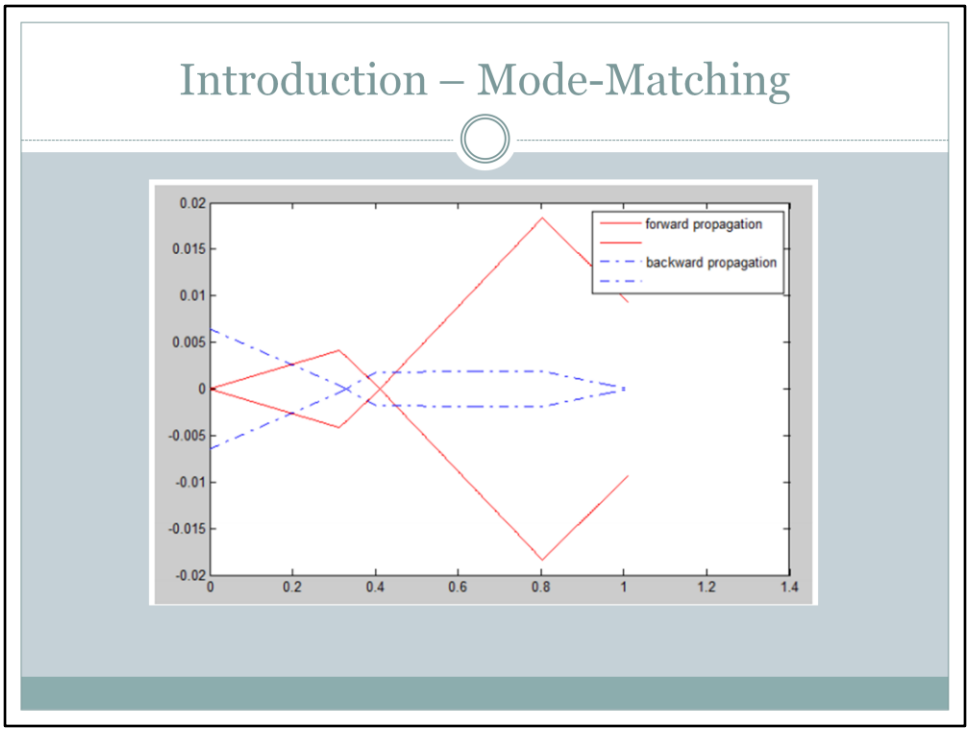
$$\frac{1}{q_2} = \frac{C + \frac{D}{q_1}}{A + \frac{B}{q_1}}$$

When we analyze a system we examine the input and output beams as affected by a particular optical element or combination of elements. Here we have the parameter q1 and q2 representing input and output beams respectively. Expanding the equation to components and dividing we are able to cancel out k the normalization factor to obtain the general form shown below.

Introduction – MATLAB

- MATrix LABoratory

As one can imagine, we would expect to use and manipulate a great number of matrices. Thus, we chose to work in a software aptly named Matrix Laboratory, or MATLAB for short. It allows us to program and simulate the propagation of Gaussian beams as well as incorporate how the beams would interact with various optical elements.

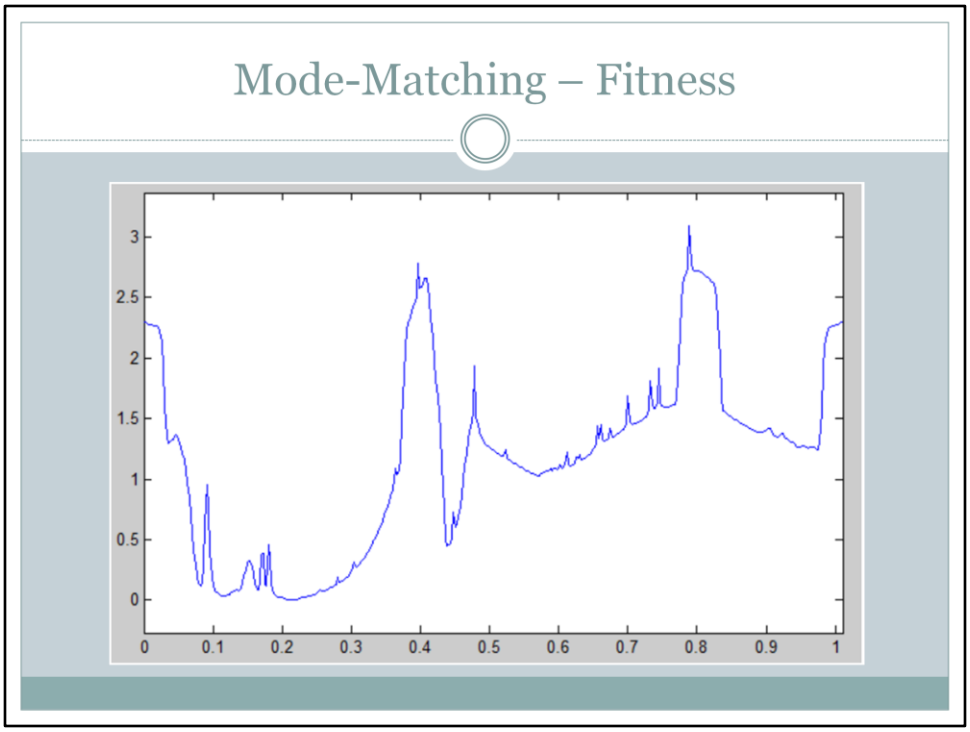


Now that we know how to manipulate Gaussian beams, the question then becomes, well, “How do we use them?”. These vary on a case by case basis for each experiment but let’s say for example we want to inject a pump laser beam into a laser cavity. To do that we would need to make sure that the pump laser matches the resonator of the previous laser to prevent undesired processes from occurring such as interference.

Previously, mode-matching was done manually. A general familiarity with lenses and optics allowed for experimental placement of lenses to make an educated guess towards what should give the output beam desired. However, we can automate this process to great efficiency and remove the need to manually mode-match.

Mode-Matching – The Process

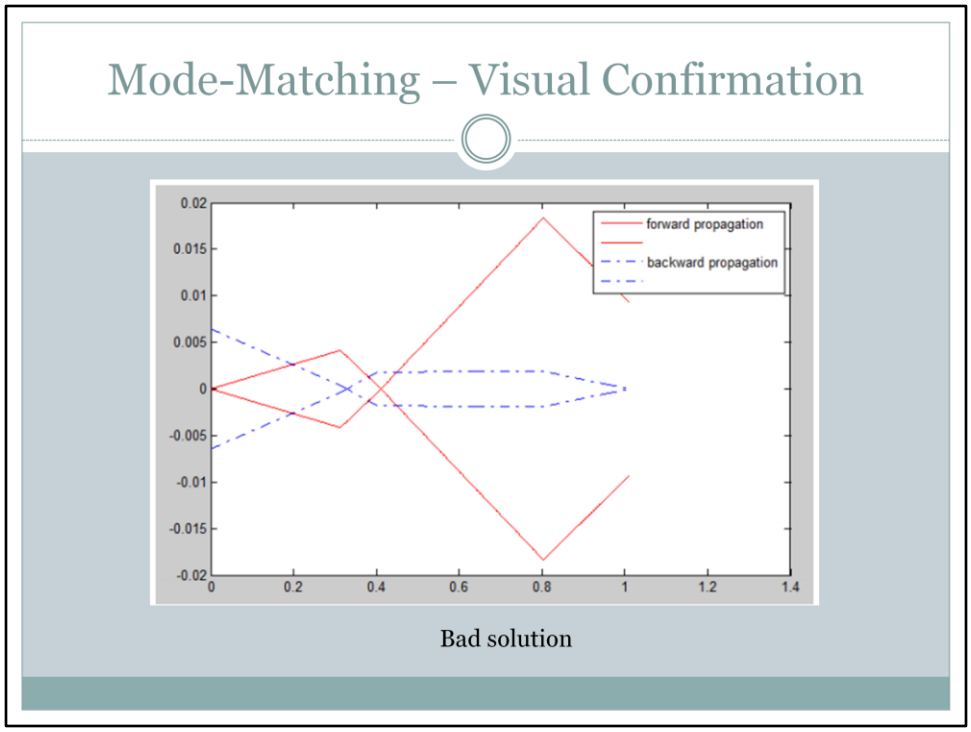
- Working off of Professor Mikhailov's Gaussian Beam Propagation code
- Old Code:
  - Accepts input parameters for initial beam and defined positions of lens with certain focal lengths
  - Plots and profiles the gaussian beam
- New Code:
  - Permutes positions of lenses
  - Checks each permutation against a "fitness"
  - Outputs 3 lens solutions



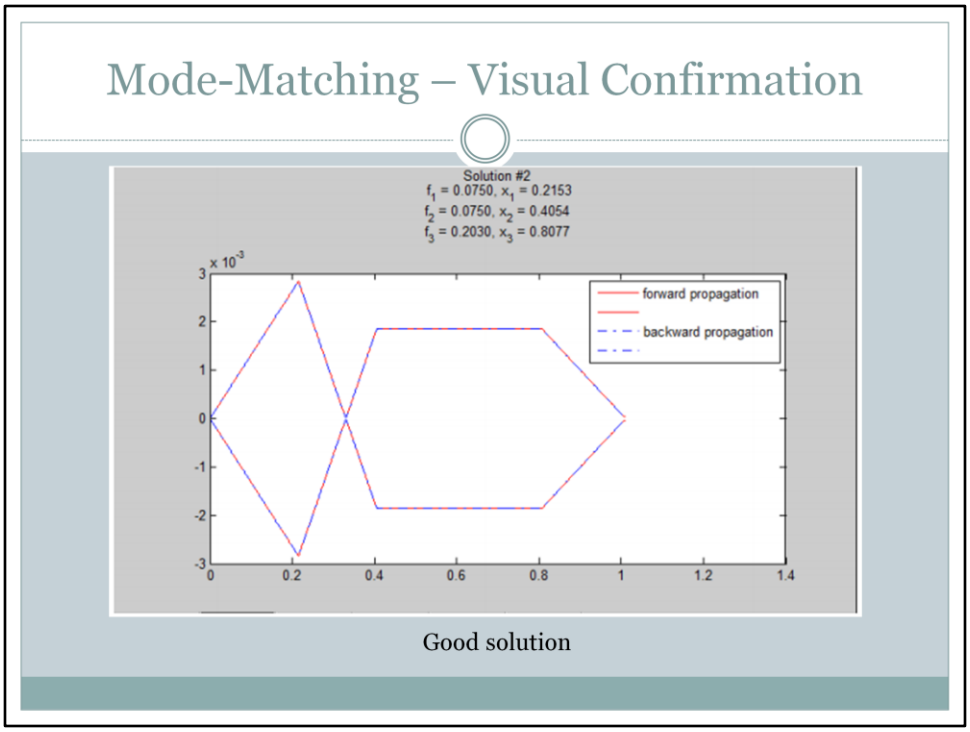
Here is a sample of what the fitness function looks like if we hold the 2nd and 3rd lens constant.

Fitness parameters:

- Not too close together
- Within edge boundaries
- Collimated region between 2nd and 3rd lens
- Forward and backward propagation match waists

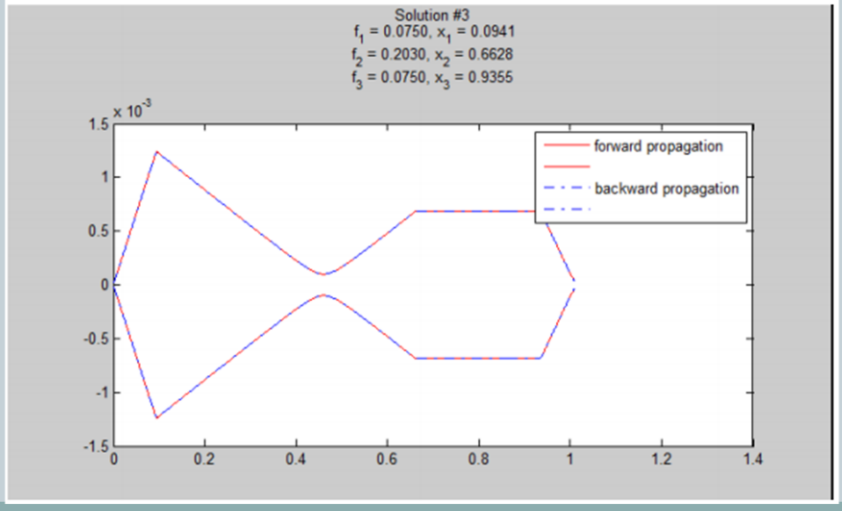


The last factor, matching waists in the forward and backward propagation of the beam, is important because it allows us to visually confirm that a beam is successfully mode-matched. If the forward and backward propagation do not match then the sensitivity of the q parameter to small deviations is apparent. However, if they do match then we do know that it is a good solution that won't deviate from the current solution if we, say, move one lens in one direction. The above picture shows an example of a bad solution, the red line representing forward propagation and the blue line representing backward propagation. The y-axis is the waist from the optical axis and the x-axis is position in meters.



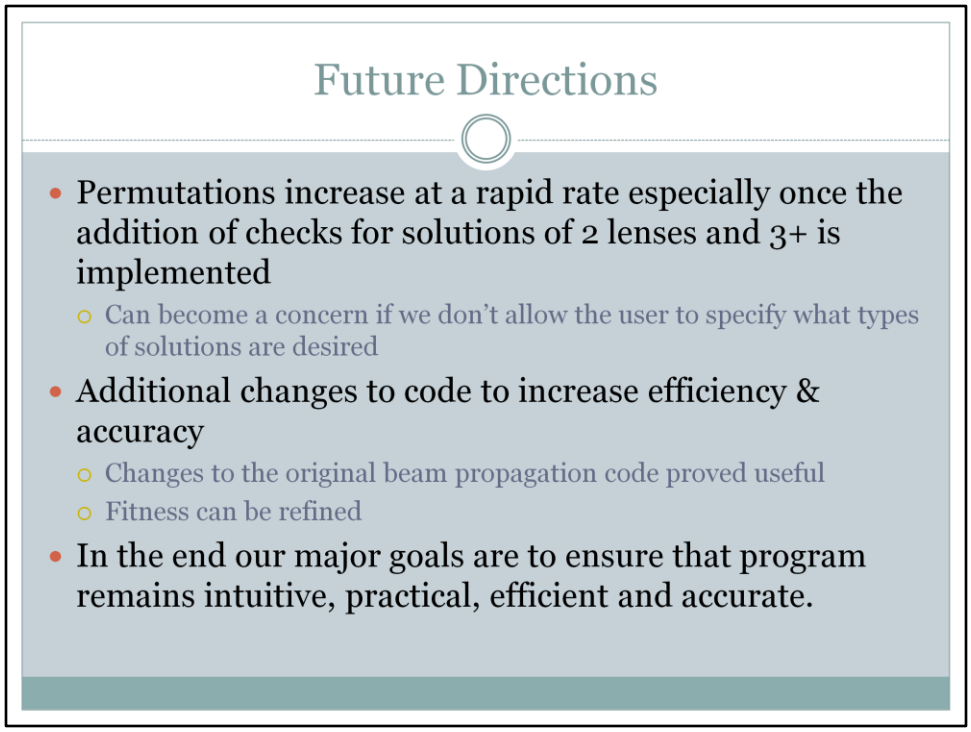
And on the other side, here is a solution that is considered good. The forward and backward propagation overlap seemingly perfectly, the lenses (at approx. .2, .4, and .8) are within the bounds of 0 to 1, the lenses are not overlapping each other, and there is a collimated region between the 2nd and 3rd lens.

Mode-Matching – Current Status



Mode-Matching – Current Capabilities

- Accepts any thin lens focal length inserted
- Permutes all possible 3 lens solutions
- Picks out unique solutions with ability to set threshold for "uniqueness"
- Plots user inputted number of solutions and outputs specifics about lens focal length and position
- Outputs final waist and radius to compare against desired output parameters
- Current runtime ~9 seconds with a lens set containing 2 lenses of different focal lengths



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.