Mirror Designer User's Guide:

Using a Genetic Algorithm to Design Multilayer Mirrors

By

Jeffery Hallstrom

Submitted to the Department of Physics and Astronomy in partial fulfillment

of graduation requirements for the degree of

Bachelor of Science

Brigham Young University

July 2005

Advisor: R. Steven Turley	Thesis Coordinator: Branton Campbell
Signature:________________________	Signature:________________________

Department Chair: Scott D. Sommerfeldt

Signature: ________________________

Table of Contents

List of Figures

1. INTRODUCTION

1.1 Purpose

The purpose of this guide is to help the user understand and effectively use the Mirror Designer program. Mirror Designer is a Java-based computer program that uses a genetic algorithm to design multilayer mirrors that optimize reflectance at wavelengths specified by the user. The genetic algorithm used by Mirror Designer is described, as well as the method used to calculate the reflectivity of multilayer mirrors.

1.2 Acknowledgments

I would like to thank Dr. Turley for his invaluable help. He presented me with the idea for the program, and aided me in finding solutions to many questions and problems along the way. This project would not have been possible without the work done by Shannon Lunt in developing the idea of using a genetic algorithm to optimize multi layer mirror design. I would also like to thank Amy Baker for her help in testing the program.

1.3 Motivation

The IMAGE Mission required the design of a mirror that would be highly reflective to light with a wavelength of 304 Å and non-reflective to 584 Å light. Shannon Lunt wrote a program using a genetic algorithm to design aperiodic multilayer mirrors to solve this problem. Shannon Lunt's program yielded several surprising mirror designs for the IMAGE Mission problem[1]. However, her program was written specifically for this one application.

Mirror Designer expands upon the work done by Lunt by creating a tool useful for solving a range of problems. Mirror Designer allows the user to specify the conditions of the problem. Mirror Designer used the IMD Optical Constants database allowing it to access data for many more materials, and for a wide range of wavelengths. A user interface was added to eliminate the steep learning curve for using the program. No computer programming knowledge is required to use Mirror Designer. The user can decide to design periodic or aperiodic mirrors. Mirror Designer also allows the user to tune the genetic algorithm by changing variable that control the process. A tolerance function was added allowing the user to test how small changes in the thickness of a mirror’s layers affects the reflectance. Mirror Designer can also display graphs of the reflectance of mirrors as well as the data in the IMD database.

1.4 Java

Mirror Designer was written in the Java. Although other languages such as C and FORTRAN are much faster, Java was selected for the following reasons. The user interface could be designed easily using Java. The program can also be made into an applet and then be posted on a webpage. This will allow many people access to Mirror Designer without needing to distribute it. Also, when changes are made to the program the webpage will be updated instead of updating all distributed copies. Java is portable between operating systems, so Mirror Designer can be run on a machine using Windows or Linux. Several other members of the X-Ray Optics Research group have used Java to develop other programs. Thus, Java was also selected for the interest of code comparison and sharing.

1.5 JMSL

Mirror Designer uses the JMSL Numerical Library for Java Applications to display graphs, interpolate data and perform local optimization. If you are running Mirror Designer locally you will need a license to access these functions. Without a JMSL license Mirror Designer will still run, however some of the above mentioned functions will not be operational. JMSL was developed by Visual Numerics. More information can be found at the following web page: http://www.vni.com/products/imsl/jmsl/jmsl.html. Mirror Designer currently uses the JMSL license issued to BYU.

If you run Mirror Designer locally and have your own license you wish to use, or need to change the license for some other reason, it can be done by changing *MirrorDesigner.bat* (See **1.6 Installation Instructions** for more information on this file). Right click on the file and select edit. The file contains one line:

java -Dcom.imsl.license.path=@*LICENSE* MirrorDesignerPkg.MirrorDesigner.

Enter the license name where *LICENSE* is written in the line above.

1.6 Installation Instructions

Unzip the file *MirrorDesigner.zip*, and place the folder *Mirror Designer* in whatever directory you choose. The folder contains 3 items:

1. The folder *com* contains the JMSL libraries.
2. The folder *MirrorDesignerPkg* contains the Mirror Designer *.class* files.
3. *MirrorDesigner.bat* is the executable file to launch Mirror Designer. You can make a short cut to this file and place it on the desktop, or wherever you choose.

Two other files were also unzipped:

1. The folder *nk.dir*, which contains the IMD database. You can place this folder wherever you choose.
2. The file *Mirror Designer Guide.pdf* is the user's guide.

Mirror Designer requires the Java Runtime Environment (JRE) or Java Development Kit (JDK) to run. They can be downloaded from http://java.sun.com/. Mirror Designer was written using JDK 5.0. The JRE or JDK must be in the PATH. You can check this by going to the command prompt: click start → run, then type “cmd”. At the command prompt type “java”. If an error message is displayed you must update the PATH to include the JRE or JDK. Instructions on doing this can be found on the internet by searching “set path”.

1.7 Getting Started

Mirror Designer has two sections: the Project section, and the Mirrors section. In the Project section you specify what you want the mirror to be by

1. Defining general attributes of the mirror such as number of layers, and a range of thicknesses.
2. Adding conditions telling Mirror Designer what wavelengths of light you want to reflect.
3. Selecting materials that Mirror Designer will consider for designing the mirror.
4. Making desired changes to advanced variables. This step is not necessary, but can allow the experienced user to control the speed and accuracy of the program.

When you run Mirror Designer it will compile a list of mirrors that best solve the problem and go to the Mirrors section. In the Mirrors section you can

1. Graph the reflective properties of the mirrors.
2. Test the mirrors’ tolerance to small changes in thickness.
3. Perform local optimization on the mirrors.

There is a text area to the right of the main panel. This text area allows Mirror Designer to give you feedback and display errors. If you try to perform an action improperly Mirror Designer will tell you what you are doing wrong. Go ahead and try running the project as soon as you open Mirror Designer and you will see a list of reasons why the project could not run.

2. THEORY

2.1 Local vs. Global Optimization

When designing multilayered mirrors there are many parameters to fit: the materials, and the thickness of each layer; thus the solution space of possible mirror is enormous. Using a local optimizer for these problems is out of the question because it would be entirely sensitive to the initial guess.

Global optimizers, such as genetic algorithms, sample much more of the solution space and are much more likely to locate the global extreme. Shannon Lunt determined that by using a genetic algorithm "the randomness in movements and in the initial population allow global extrema to be found.[1]" However, global optimization is much more time intensive than local optimization.

In essence the solution space can be thought of as a multi-dimensional surface with many peaks and valleys. The mirror that best satisfied the problem lies at the tallest peak, the global maximum. A local optimizer makes an initial guess and goes up hill until it reaches a peak. However this peak is not the global maximum; it is the closest peak to the initial guess. A global optimizer makes many initial guesses. Then by making educated and random changes to the parameters finds the global maximum. Once the global optimizer has found the "tallest mountain" it does not go directly to the peak but continues to sample other "mountains" while slowly working its way up. Once the "tallest mountain" has been found the local optimizer is much faster. The quickest way to find the global maximum is to use the genetic algorithm to generate starting positions for the local optimizer.

2.2 The Genetic Algorithm

Genetic algorithms receive their name because of their similarities to the natural process of survival of the fittest. Genetic algorithms can be applied to many types of problems. This section describes the genetic algorithm used in Mirror Designer. The genetic algorithm is a modified version of the algorithm used in Lunt's program. Some of the differences are discussed below.

2.2.1 Merit Function

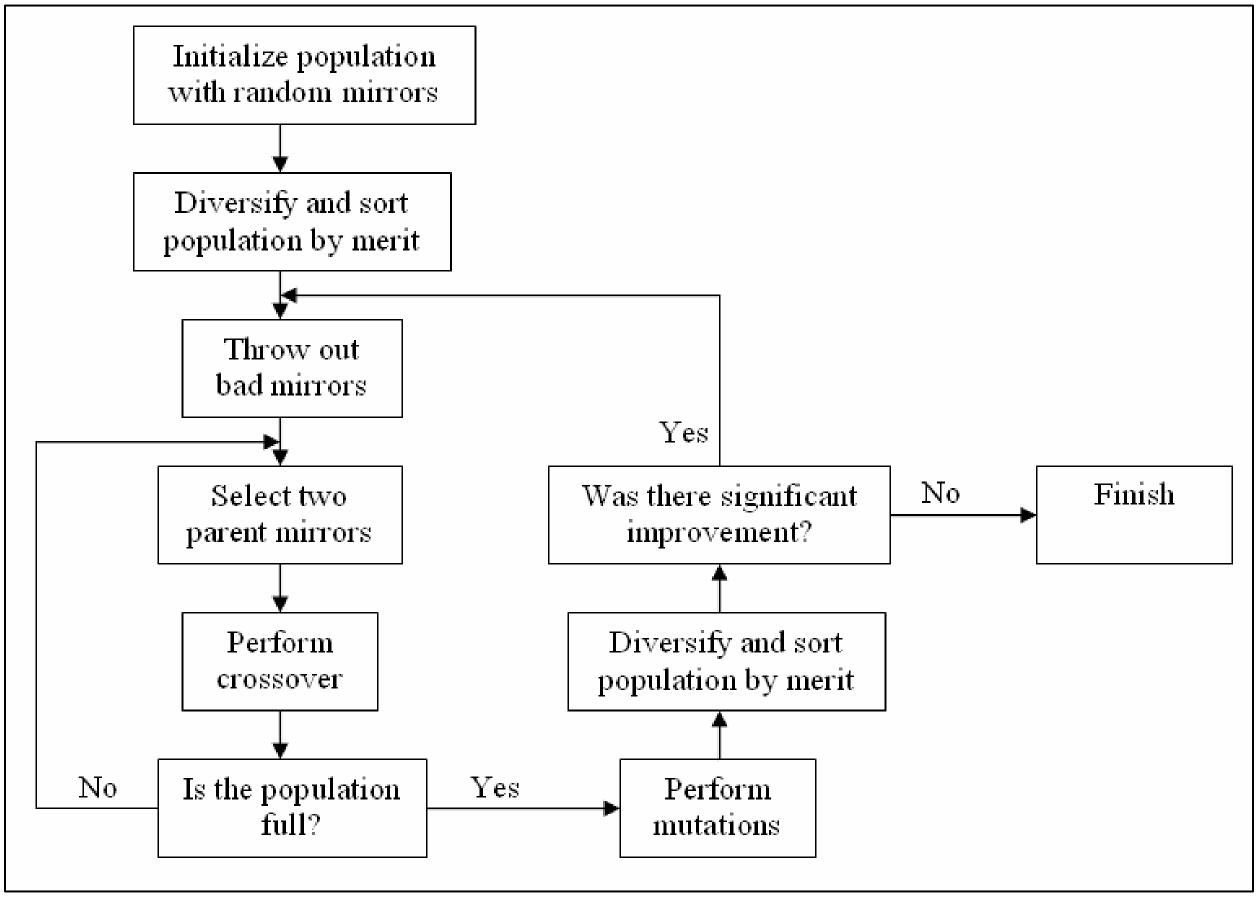
In order to judge how well, or how poorly, a possible mirror meets the requirements, it is useful to define a merit function for the problem. A merit function assigns a number to a mirror depending on how well it fulfills the needs of the problem. The best mirror for the project will be the mirror with the highest merit value. The genetic algorithm is used to search through the solution space of possible mirrors and find the mirror with the highest merit value.

Mirror Designer has three different merit functions which are described in the user's guide.

2.2.2 Process

The attributes of the first generation of mirrors is chosen at random. Then the merit value for each mirror is calculated and the set of mirrors is sorted. The mirrors with high merit value are kept, and the mirrors with low merit values are eliminated. Parent mirrors are chosen from those mirrors with high merit values and their attributes are combined to create children mirrors. Then a new generation is made containing the children mirrors and the best mirrors from the previous generation. Some mirrors are altered randomly each generation, similar to mutations in a species.

This process is repeated until the merit values of the best mirrors do not improve significantly. At this point the mirrors with the absolute best merit values have been found. There are many attributes of the genetic algorithm that can be changed to alter speed and accuracy of the algorithm: population size, percentage of each generation to be replaced, probability of mutation, etc. These are discussed in more detail in **3.4 Advanced**.


Initialize population with random mirrors
Diversify and sort population by merit
Throw out bad mirrors
Select two parent mirrors
Perform crossover
Is the population full?
No
Yes
Perform mutations
Diversify and sort population by merit
Was there significant improvement?
Yes
No
Finish
Figure 1: The genetic algorithm

2.2.3 Crossover

The attributes of each mirror can be compared to a *chromosome* carrying the DNA of an organism. Each attribute of the mirror can be compared to a single *allele*. Thus a six layered mirror containing two materials can be represented by a chromosome containing eight alleles: one for each material, and one for the thickness of each layer.

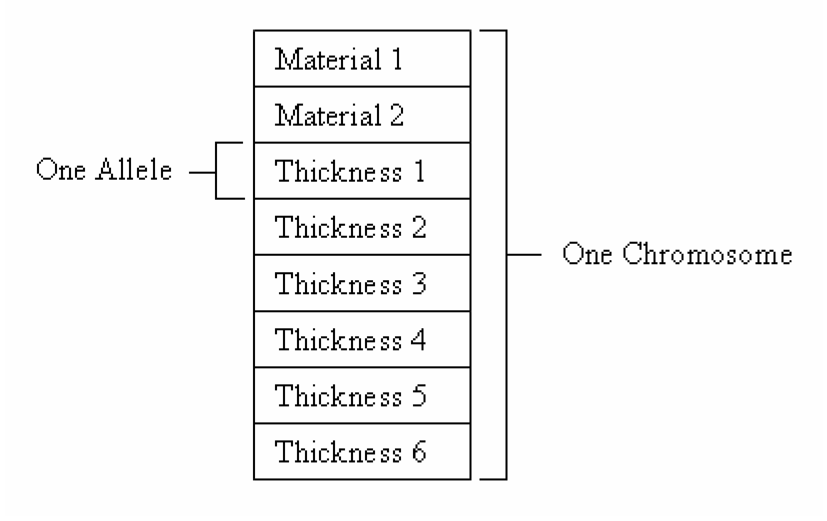


Figure 2: Chromosomes and alleles

When a crossover is performed two parent mirror chromosomes are combined to make two children mirror chromosomes. A crossover point is randomly selected. The first child gets alleles from the first parent up to the crossover point, and it gets alleles from the second parent after the crossover point. The second child gets alleles from the second parent up to the crossover point, and it gets alleles from the first parent after the crossover point.

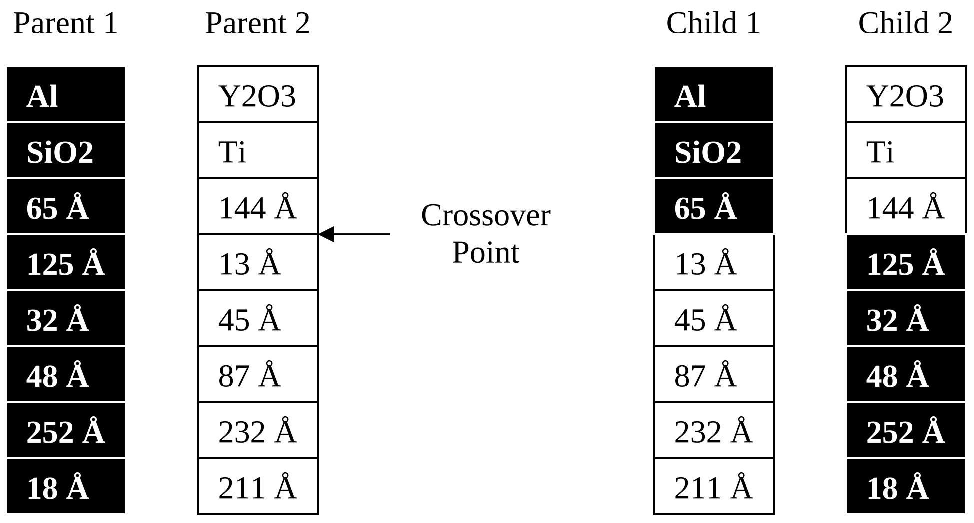


Figure 3: Crossover

2.2.4 Mutation

Mutations are done by randomly choosing an allele, and changing it to a new random value. This is different than the way mutations are performed in Lunt's program. Lunt stored each allele in a byte. Then when mutating an allele one bit was changed from 0 to 1, or 1 to 0. Lunt's method more accurately depicts the way mutation occurs in nature; however, a byte can store a number between 0 and 255, thus using one byte to store each allele creates restraints on the range of thicknesses. When testing Mirror Designer against Lunt's results, no significant different was seen. The two methods are similar enough to yield the same end result.

2.3 Absorption

When the merit function of a multilayer mirror is calculated we are required to calculate its reflectance. At each interface some light is reflected while some light is transmitted to the next layer. When the index of refraction is complex some light is also absorbed[2]. In the extreme ultraviolet region the index of refraction is often complex. Equation (1) describes a plane wave.

$$\vec{E}(\vec{r},t) = \vec{E}_o e^{i\left(\vec{k}\cdot\vec{r} - \omega t + \phi\right)}, \tag{1}$$

where $\vec{k}$ is the wave vector, and ϕ represents a real phase. The wave vector in a material is defined as

$$\vec{k} = \frac{2\pi \mathrm{N}}{\lambda_{vac}}\hat{u}, \tag{2}$$

where N is the index of refraction. A complex index of refraction is given by

$$\mathrm{N} = n + i\kappa, \tag{3}$$

where both n and κ are real. If N is complex then $\vec{k}$ becomes

$$\vec{k} = \frac{2\pi}{\lambda_{vac}}(n + i\kappa)\hat{u}\,, \tag{4}$$

and $\vec{E}$ becomes

$$\vec{E}(\vec{r},t) = \vec{E}_o e^{\frac{-2\pi\kappa}{\lambda_{vac}}\hat{u}\cdot\vec{r}} e^{i\left(\frac{2\pi n}{\lambda_{vac}}\hat{u}\cdot\vec{r} - \omega t + \phi\right)}. \tag{5}$$

Assuming normal incidence

$$\hat{u} = \hat{z}\,, \qquad \vec{k} = \frac{2\pi\mathrm{N}}{\lambda_{vac}}\hat{z}\,, \tag{6}$$

and

$$\vec{E}(\vec{r},t) = \vec{E}_o e^{\frac{-2\pi\kappa}{\lambda_{vac}}z} e^{i\left(\frac{2\pi n}{\lambda_{vac}}z - \omega t + \phi\right)}. \tag{7}$$

$e^{\frac{-2\pi\kappa}{\lambda_{vac}}z}$ is the damping term, and the $e^{i\left(\frac{2\pi n}{\lambda_{vac}}z - \omega t + \phi\right)}$ term can be written as

$$\cos\left(\frac{2\pi n}{\lambda_{vac}}z - \omega t + \phi\right) + i\sin\left(\frac{2\pi n}{\lambda_{vac}}z - \omega t + \phi\right). \tag{8}$$

The graph of the damping term and the real part of the other term, shown by Figure 4, illustrates how the electric field decays as it propagates through the material. If the index of refraction were not complex and $\kappa = 0$, there would be no damping term.

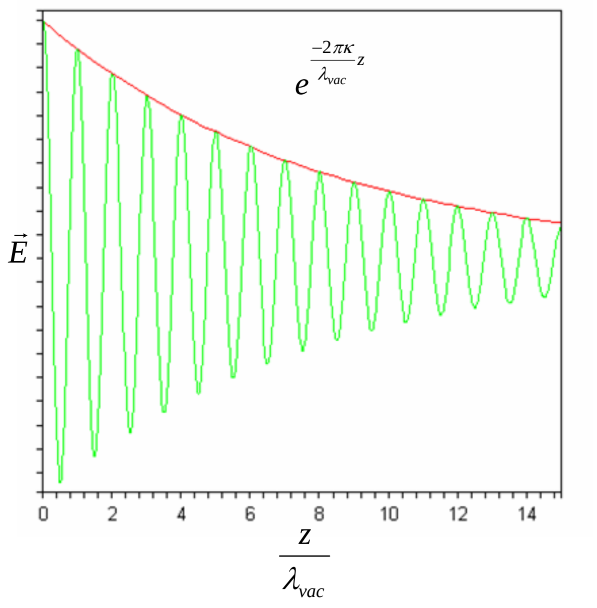


Figure 4: Decay of the electric field

2.4 Polarization and Reflectance

In Figure 5 we assume a planar interface between two materials. The two materials are characterized by their indices of refraction, n_1 and n_2. The incident plane wave($\vec{k}_i$) strikes the interface and part of the wave is reflected($\vec{k}_r$) while part is transmitted($\vec{k}_t$). The electric field of each wave can be divided into *s* and *p* components which are perpendicular to $\vec{k}$. $\vec{E}_P$ is chosen to be in the plane of incidence, and $\vec{E}_S$ is chosen to be normal to the plane of incidence.

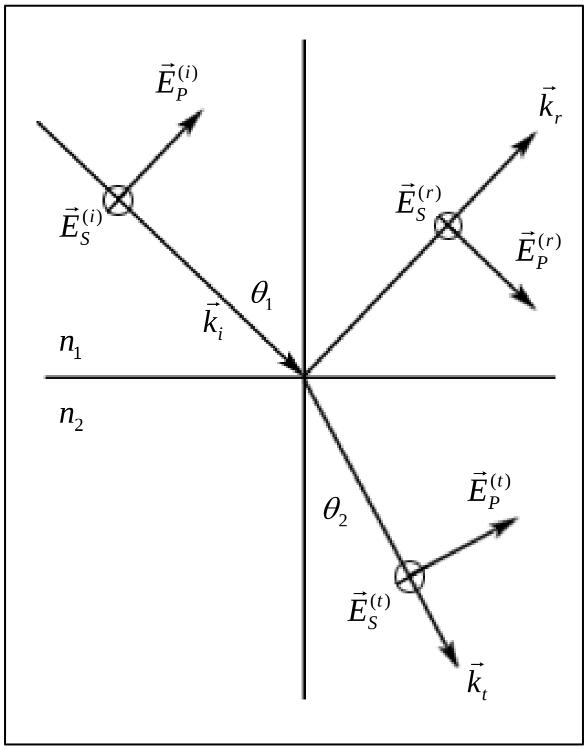


Figure 5: Reflectance and transmission of a plane wave at a planar interface

The polarization of the incident light must be considered. The problem Lunt addressed assumed unpolarized incident light, equal amounts of *s* and *p* polarizations. However, in other problems, depending on the light source, there could be any combination ranging from completely *s* polarized light to completely *p* polarized light.

We are interested in finding the percentage of incident light that is reflected. The initial intensity is given by

$$I_{total}^{(i)} = I_S^{(i)} + I_P^{(i)} . \tag{9}$$

The reflected intensity is given by

$$I_{total}^{(r)} = R_S I_S^{(i)} + R_P I_P^{(i)} , \tag{10}$$

where R_S and R_P are the fractions of *S* and *P* polarized light reflected from the surface. The percentage of incident light that is reflected is then

$$\frac{I_{total}^{(r)}}{I_{total}^{(i)}} = \frac{R_S I_S^{(i)} + R_P I_P^{(i)}}{I_{total}^{(i)}} = R_S \frac{I_S^{(i)}}{I_{total}^{(i)}} + R_P \frac{I_P^{(i)}}{I_{total}^{(i)}} . \tag{11}$$

$\frac{I_S^{(i)}}{I_{total}^{(i)}}$ is the percentage of the incident light that is *S* polarized, and $\frac{I_P^{(i)}}{I_{total}^{(i)}}$ is the percentage of the incident light that is *P* polarized. These are be determined by the light source. For one interface, as shown in figure 5, R_S and R_P are given by $R_S = \left|r_S^2\right|$ and $R_P = \left|r_P^2\right|$ where r_S and r_P are the well known Fresnel coefficients. For a multilayer mirror R_S and R_P can be found using Parratt's recursive method.

2.5 Parratt's Recursive Method

This section is based upon Parratt's recursive method for calculating reflectance of multilayer

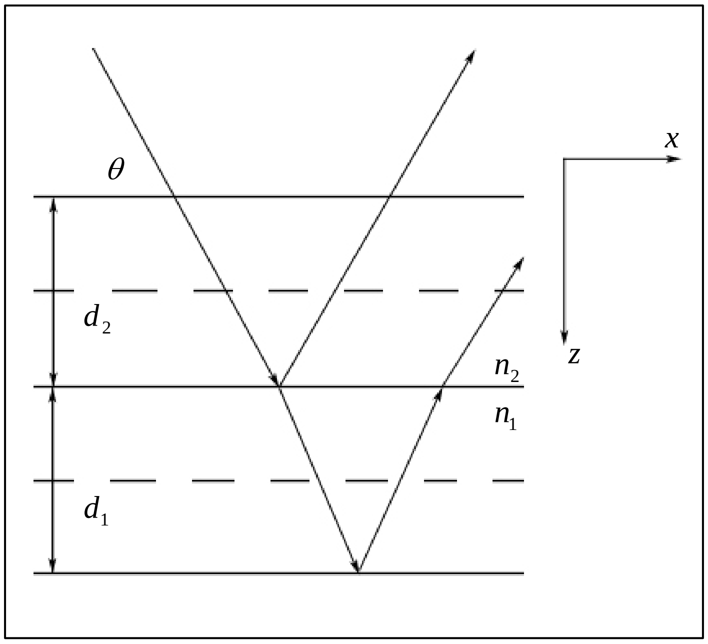


Figure 6: Geometry of Parratt's recursive method

systems as used by Kohn[3][4]. Lunt's paper also served as a resource for this section[1]. Consider a mirror composed of many thin layers of different materials, where each interface is assumed to be planar. A plane wave with wave vector $\vec{k}$ enters the mirror. At each interface part of the wave is reflected and part is transmitted into the next layer. Thus, in each inner layer there are two waves: one transmitted from the interface above and one reflected from the interface below. Both waves move in the positive *x* direction, while the transmitted wave travels in the positive *z* direction as the reflected wave travels in the negative z direction. The electric field from both waves in the *n*th layer can be given by

$$E_n(x,z,t) = E(x,t)\left[\vec{E}_{tn} e^{(ik_{zm}z)} + \vec{E}_{rn} e^{(-ik_{zm}z)}\right], \tag{12}$$

where

$$E(x,t) = e^{i(k_x x - \omega t)} \tag{13}$$

and $\vec{E}_{tn}$ and $\vec{E}_{rn}$ are the amplitudes of the transmitted and reflected waves at the middle of the layer. *Z* is also measured from the center of the layer.

The *x* and *z* components of $\vec{k}$ are

$$k_x = \frac{2\pi}{\lambda_{vac}} \cos\theta \tag{14}$$

and

$$k_z = \sqrt{\left(\frac{2\pi \mathrm{N}}{\lambda_{vac}}\right)^2 - k_x^2}\,, \tag{15}$$

where θ is the angle from grazing, and N is the index of refraction. The *x* component is independent of the index of refraction. The index of refraction may be complex: $\mathrm{N} = n + i\kappa$. Written in terms of k_z the Fresnel coefficients are

$$r_{21}^{S} = \frac{k_{z2} - k_{z1}}{k_{z1} - k_{z2}} \tag{16}$$

and

$$r_{21}^{P} = \frac{\mathrm{N}_1^2 k_{z2} - \mathrm{N}_2^2 k_{z1}}{\mathrm{N}_2^2 k_{z1} + \mathrm{N}_1^2 k_{z2}}, \tag{17}$$

where the numbers 1 and 2 refer to the layer number. We count the layers starting at the bottom of the mirror moving towards the top. So, layer 1 is below layer 2. The phase of the wave after propagating through half of layer *n* is

$$C_n = e^{\left(i\frac{k_{zn}d_n}{2}\right)}, \tag{18}$$

where d_n is the thickness of layer *n*.

We can now use the recursive equations

$$R_2'^{S} = \frac{C_2^4\left(r_{21}^S + R_1'^S\right)}{1 + r_{21}^S R_1'^S} \tag{19}$$

and

$$R_2'^{P} = \frac{C_2^4\left(r_{21}^P + R_1'^P\right)}{1 + r_{21}^P R_1'^P}, \tag{20}$$

where $R_n' = C_n^2 R_n$. The reflectance of each layer depends upon the reflectance of the layer below it. Equations (19) and (20) are applied recursively starting at the bottom of the mirror, where R_2 is the lowest layer and $R_1 = 0$ because there is no reflected wave in the lowest layer. Then the result is plugged back into the equation as R_1 now represents the lowest layer and R_2 is the layer above it. This recursive process is repeated until R_1 is the top layer and R_2 is the air, or vacuum, above the mirror. R is usually complex in the extreme ultraviolet range, so the physical reflectance of the multilayer mirror is given by

$$R_{total}^{S} = \left|R^{2}\right| \tag{21}$$

and

$$R_{total}^{P} = \left|R^{P}\right|. \tag{22}$$

R_{total}^{S} and R_{total}^{S} can now be plugged in for R_S and R_P in equation (11) to give the percentage of the incident light that is reflected by the multilayer mirror.

3. PROJECT SECTION

The Project section has four panels: **General**, **Materials**, **Output**, and **Advanced**. The tabs at the top of the page allow you to move from one panel to another.

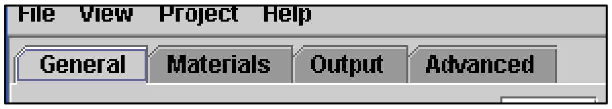


Figure 7: Project tabs

3.1 General

In the general panel you tell Mirror Designer what you want the mirror to do.

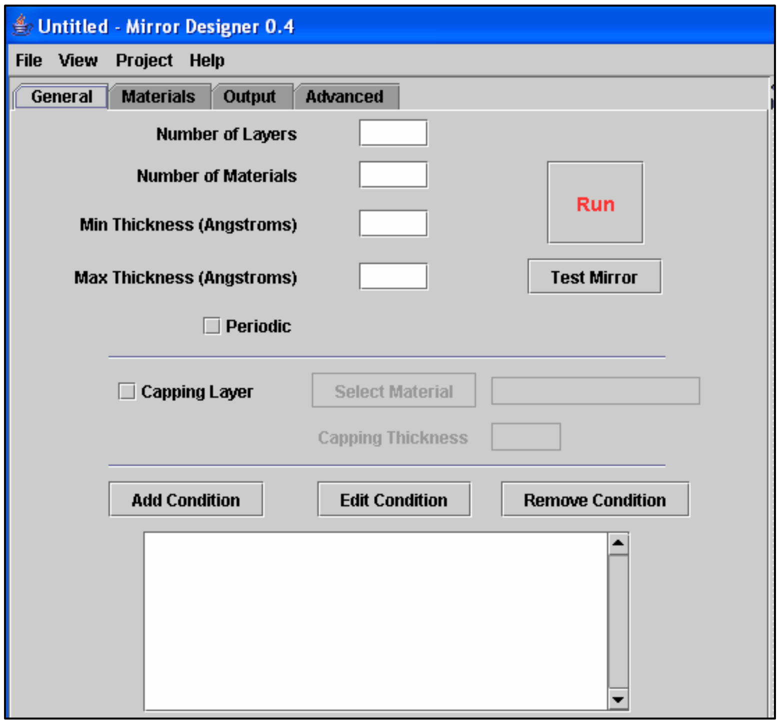


Figure 8: General Panel

Number of Layers - The number of layers the mirror will have.

Number of Materials - The number of different materials the mirror will have. This is not to be confused with the number of materials Mirror Designer will consider when designing the mirror. A mirror with 6 layers and 2 materials is shown in figure 9. If more than three materials are used, a very high Population Size will be needed and the project will run for an extremely long time. See Population Size in **3.4 Advanced** for details.

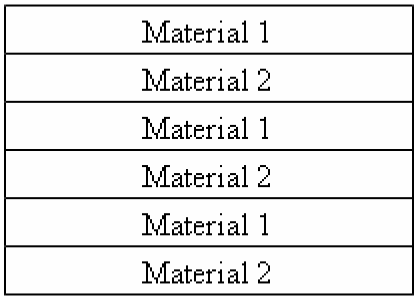


Figure 9: A mirror with 6 layers and 2 materials

Min Thickness - Each layer will have at least this thickness. Thicknesses are measured in Angstroms.

Max Thickness - Each layer will have a thickness no larger than this number. Thicknesses are measured in Angstroms.

Periodic - If checked, Mirror Designer will only generate periodic mirrors.

Run - Click this button to run the project.

Test Mirror - This button allows you to test a specific mirror. Mirror Designer will ask for the materials and thicknesses at each layer then go to the Mirrors section. If you want a capping layer or a periodic mirror you must check the boxes before clicking the Test Mirror button.

3.1.1 Capping Layer

Many of the materials used in multilayered mirrors will oxidize over time if left exposed. If the top layer of the mirror oxidizes it can dramatically change the reflective properties of the mirror. To prevent the top layer from oxidizing you can specify a capping layer for the mirror. You should choose a material that will not oxidize, generally an oxidized compound. You must also specify the thickness of this capping layer.

When you click the Select Material button you must have previously located the database of materials.

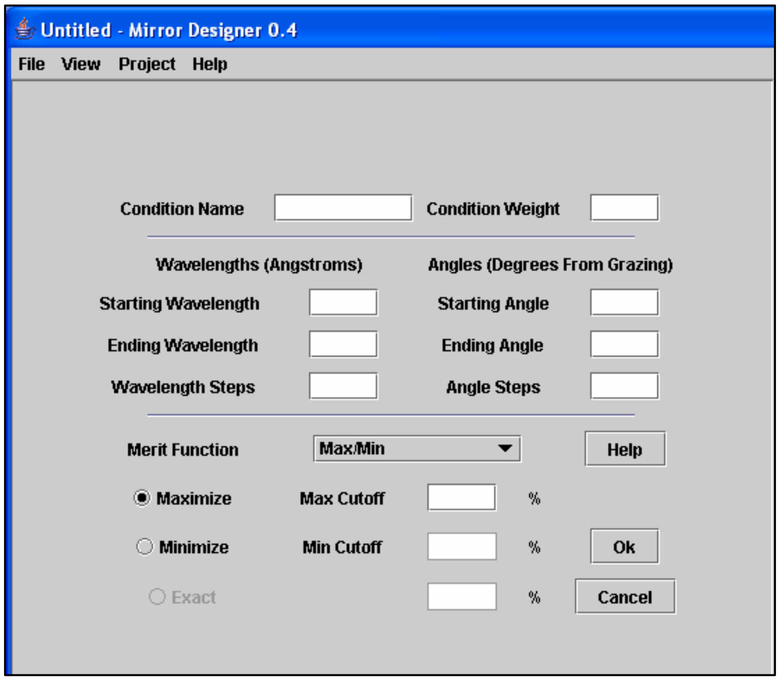
3.2 Conditions

Mirror Designer uses conditions to allow you to specify the reflective properties you want the mirror to have. You can create many conditions, and they will all be displayed in the general section below the condition buttons. The order in which you create conditions does not mater.

Condition Name - The name exists only to help you remember the condition.

Condition Weight - A condition's weight tells Mirror Designer how much to emphasize this condition compared to others. For example say you create two conditions (conditions *A* and conditions *B*). Say *A* is more important, or harder to satisfy, than *B*. You can give *A* a weight of 2 and *B* a weight of 1. It is the ratio that is important. You could use 4 and 2, or 10 and 5 and it would make no difference. Also, if you have only one condition, it makes no difference what weight you use. See **3.2.1 Merit Functions** for more information.

Wavelengths - If the condition is only interested in one wavelength enter it into the Start Wavelength field. Leave the other two fields empty. If the condition is interested in a range of wavelengths enter the starting, ending wavelengths and the number of steps in-between. The number of steps does not count the starting wavelength as a step. For example if you enter 300, 304 and 4, the condition will consider 300 Å, 301 Å, 302 Å, 303 Å and 304 Å. Wavelengths are measured in Angstroms.



Merit Function: Max/Min
[✓ Maximize] Minimize Exact

Figure 10: Condition window

Angles - If the condition is only interested in one angle enter it into the Start Angle field and leave the other two fields empty. If the condition is interested in a range of angles enter the starting angle, ending angle and number of steps in-between. The number of steps does not count the starting angle as a step. For example with a starting angle of 30° and an ending angle of 32° and angle steps is set to 4, Mirror Designer will look at the following angles: 30°, 30.5°, 31°, 31.5° and 32°. Angles are measured in degrees from grazing.

Number of Steps - The number of steps you specify for the wavelengths and angles affects the speed of the genetic algorithm. Each time the merit function for this condition is calculated the reflectance at each wavelength and angle must be calculated. If you specify 100 Wavelength Steps and 100 Angle Steps, then 10,000 reflectances must be calculated for each mirror. Generally there are thousands of mirrors in each generation, and the genetic algorithm runs for several generations. You can see that with the number of steps set too high the genetic algorithm will progress very slowly. Unless there is some irregularity in the range of data you are stepping through, a high number of steps will not yield a different result than a reasonable number of steps.

Polarization – Enter a number between -1 and 1. -1 indicates *s* polarized light, 0 indicated unpolarized light, and 1 indicates *p* polarized light. See **2.4 Polarization and Reflectance** for more information.

3.2.1 Merit Functions

A Merit Function is an algorithm for determining how well a specific mirror satisfies a condition. The best mirror will be the mirror with the highest Merit Value. There are several different merit functions that Mirror Designer can use.

Max / Min - The “Max divided by Min” merit function was used by Shannon Lunt’s program. Essentially, the average reflectance of the wavelengths you want to maximize go in the numerator, and the average reflectance of the wavelengths you want to minimize go in the denominator. If you have a mirror that reflects 22.06 % of 304Å light, and 0.219 % of 584Å light the merit value would be 22.06 / 0.219 = 100.73.

This merit function tends to emphasize minimization because as the denominator approaches 0 the Merit Value blows up. If left unchecked this merit function would ignore the numerator and focus only on minimizing the denominator. The Minimum Cutoff can be used to solve this problem. If you enter .2 % as the Minimize Cutoff / Reflectance, then the denominator will be .2 even if the average reflectance of 584Å is less than .2 %.

```
If (Min > Cutoff)
    MeritValue = Max / Min
If (Min < Cutoff)
    MeritValue = Max / Cutoff
```

The Maximum Cutoff works the same way, but is not nearly as important for this Merit Function. Generally it is set to 100 %. If you set it to 40%, and the Max reflectance is higher than 55 %, the merit value will be calculated with 40 instead of 55.

This merit function is different from the others in that the merit value of each condition is not independent of the other conditions. The Max and Min reflectance percentages come from two different conditions. If only Maximize conditions are made the denominator is set to 1. If only Minimize conditions are made the numerator is set to 1.

```
If (Min = null)
    MeritValue = Max / 1
If (Max = null)
    MeritValue = 1 / Min
```

This merit function does not support conditions trying to meet an exact reflectance.

Add – Subtract (1) - The “Add and Subtract (1)” merit function is the simplest to use. If the condition is to maximize reflectivity, the reflectance is the merit value for this condition. If the condition is to minimize reflectivity, then the negative of the reflectance is the merit value for the condition. If the condition is to meet an exact reflectivity, the negative of the absolute value of the difference from the average reflectance to the desired reflectance is the merit value.

```
If (Max)
        Merit Value = Reflectance
If (Min)
        Merit Value = - Reflectance
If (Exact)
        Merit Value = - | Reflectance - Desired Reflectance |
```

Thus the highest total merit value will have the largest Max reflectance, the smallest Min reflectance, and the smallest difference from the reflectance to the desired reflectance. This merit function does not emphasize minimization as much as the Max / Min merit function. When using this function it is often helpful to give minimizing conditions a higher weight.

The Cutoffs aren’t useful to this merit function. The Max should be set to 100% and the Min to 0%.

Add – Subtract (2) - This merit function is similar to the Add – Subtract (1) merit function. If the condition is to maximize reflectivity, the merit value is the negative of the amount below the target reflectance. If the condition is to minimize reflectivity, then the merit value is the negative of the amount the reflectance is above the target reflectance. If the condition is to meet an exact reflectivity, the negative of the absolute value of the difference from the average reflectance to the desired reflectance is the merit value.

```
If (Max)
    Merit Value = - (Max Target – Reflectance)
If (Min)
    Merit Value = - (Reflectance – Min Target)
If (Exact)
    Merit Value = - | Reflectance – Desired Reflectance |
```

The merit values are usually negative. This isn't a problem; the best mirror will still have the largest, least negative, merit value. If you get a reflectance that is past the target reflectance, the merit value becomes positive.

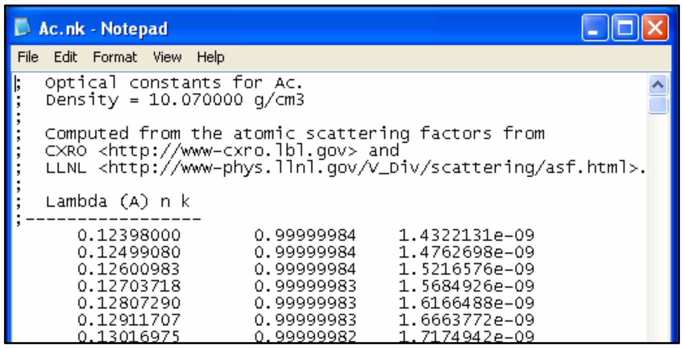
This merit function actually is very similar to the Add – Subtract(1) merit function. The only difference is in the maximize case, and the result seems to be the same. The Add – Subtract(1) merit function is the same as running the Add – Subtract(2) merit function with the target reflectance always set to 0. This merit function was made to allow the user to specify an accurate target reflectance and not need to weight minimizing conditions. However, in practice it does not seem to respond differently than Add – Subtract(1).

3.3 Materials

The Materials Panel allows you to select the materials Mirror Designer will consider when designing your mirror.

3.3.1 IMD Database

Mirror designer uses the IMD Optical Constants Database. It can be downloaded from the following website: http://cletus.phys.columbia.edu/windt/idl/. Look for a folder called *nk.dir*. The IMD Database is made up of files with the extension *.nk*. Each file contains information for one material. There is also a file called *AAACATALOG.TXT*, which contains a list of all materials in the database and the range of wavelengths contained in each file.



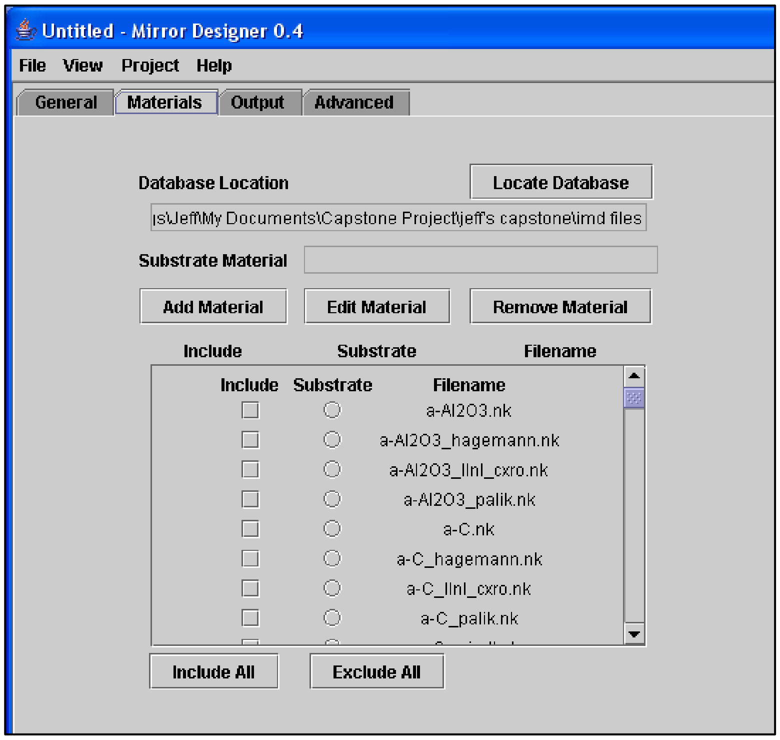
```
; Optical constants for Ac.
; Density = 10.070000 g/cm3
;
; Computed from the atomic scattering factors from
; CXRO <http://www-cxro.lbl.gov> and
; LLNL <http://www-phys.llnl.gov/V_Div/scattering/asf.html>.
;
; Lambda (A) n k
;-----------------
      0.12398000        0.99999984     1.4322131e-09
      0.12499080        0.99999984     1.4762698e-09
      0.12600983        0.99999984     1.5216576e-09
      0.12703718        0.99999983     1.5684926e-09
      0.12807290        0.99999983     1.6166488e-09
      0.12911707        0.99999983     1.6663772e-09
      0.13016975        0.99999982     1.7174942e-09
```

Figure 11: Ac.nk

Each *.nk* file contains the name of the material in the first line, several lines of notes, and the three columns of data.

1. The first column contains a list of wavelengths measured in angstroms.
2. The second column contains the *n* value for the material at the corresponding wavelength.
3. The third column contains the κ value for the material at the corresponding wavelength.

Click on the Locate Database button to specify where the database of *.nk* files is. You may choose to use a local directory or a URL containing the database. Once a valid database has been identified the Materials Panel will be contain a lengthy list of materials.


Untitled - Mirror Designer 0.4
File View Project Help
General Materials Output Advanced
Locate Database
Add Material Edit Material Remove Material
Include All Exclude All

Database Location: js\Jeff\My Documents\Capstone Project\jeff's capstone\imd files

Substrate Material:

Include	Substrate	Filename
☐	○	a-Al2O3.nk
☐	○	a-Al2O3_hagemann.nk
☐	○	a-Al2O3_llnl_cxro.nk
☐	○	a-Al2O3_palik.nk
☐	○	a-C.nk
☐	○	a-C_hagemann.nk
☐	○	a-C_llnl_cxro.nk
☐	○	a-C_palik.nk

Figure 12: Materials Panel

3.3.2 Substrate

The substrate is the material that the mirror will be built upon. You may select a substrate material by selecting the button under *substrate* for that material. The circular substrate buttons in the list of materials are radio buttons. In a set of radio buttons only one can be selected at a time.

3.3.3 Include

Each material in the list has an include checkbox that corresponds to it. When Mirror Designer runs it will design the mirrors out of these materials. It will not consider any materials without a checked include box.

Different materials have data for different ranges of wavelength. If a material is checked but it does not have data for the wavelengths in the conditions, Mirror Designer cannot run. For example, if you are trying to maximize 530 Å light and you check the include box for Actinium Mirror Designer will bring up an error message because the file *Ac.nk* only has data up to 413 Å.

If you want to use a material but it doesn't have data for the wavelengths you are interested in, you can add the data you need to the file, assuming you know the data (See **3.3.5 Adding and Editing Materials**). You can also make a new file for the material that contains the data you need. Using the above example, you can create a material called *Ac_2.nk* that contains only the data at 530 Å. This eliminates a possible problem with interpolation (see **3.3.4 Interpolation**).

3.3.4 Interpolation

Mirror Designer almost always interpolates to get data from a *.nk* file. The Interpolation can be done using linear interpolation or using JMSL's CsAkima class. Linear interpolation is like drawing a strait line between the know points and intersecting that line at the point of interest. JMSL has several methods for interpolating. CsAkima was chosen because “it keeps the shape of the data while minimizing oscillations”[5]. The different methods for interpolating are shown in Figure 13 below. The CsAkima method for interpolation yields more accurate data, but the linear interpolation is included in Mirror Designer to be used if the user does not have access to the JMSL libraries or if an error occurs in the CsAkima method.

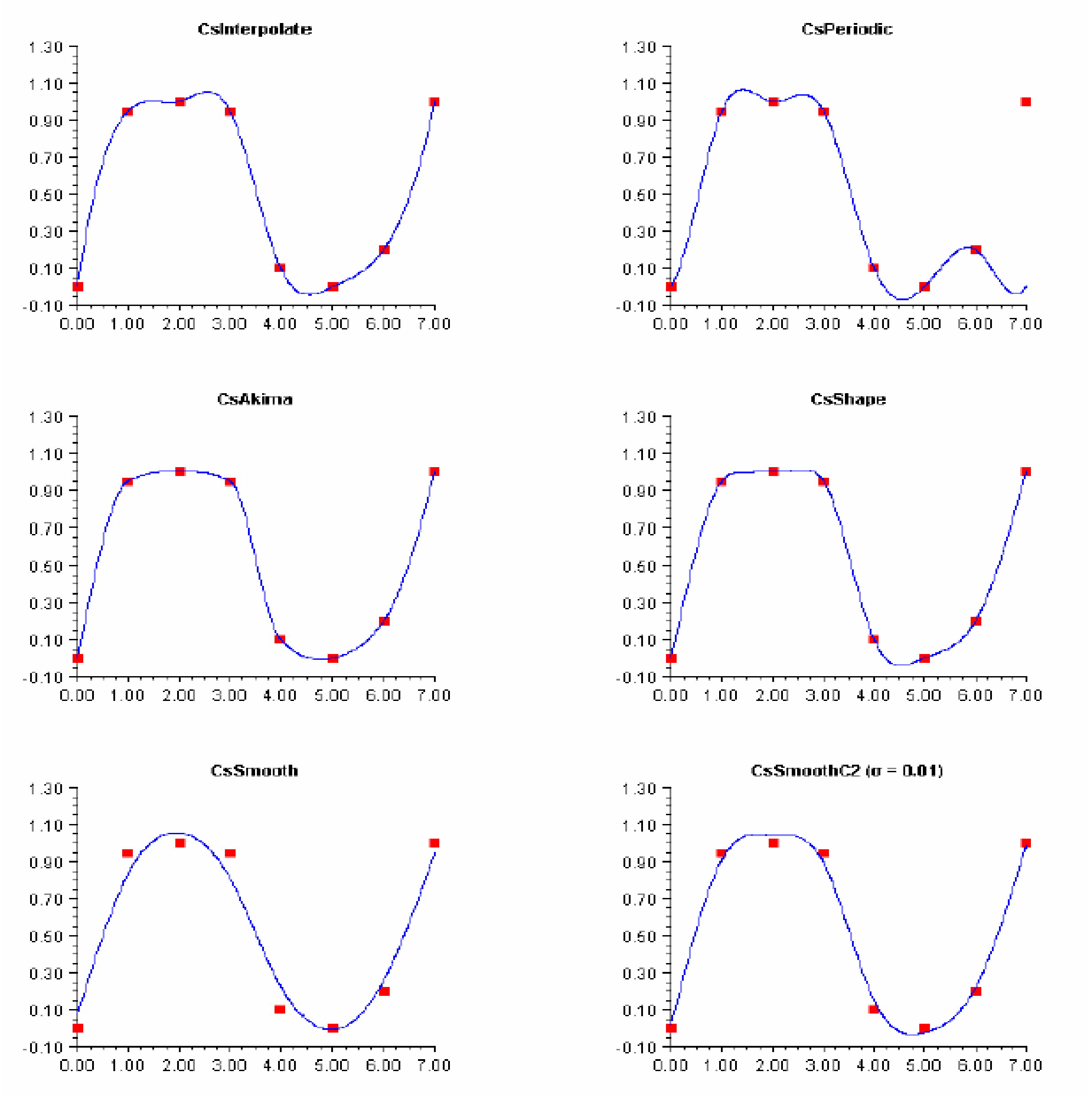


Figure 13: JMSL’s interpolation functions
borrowed from [5]

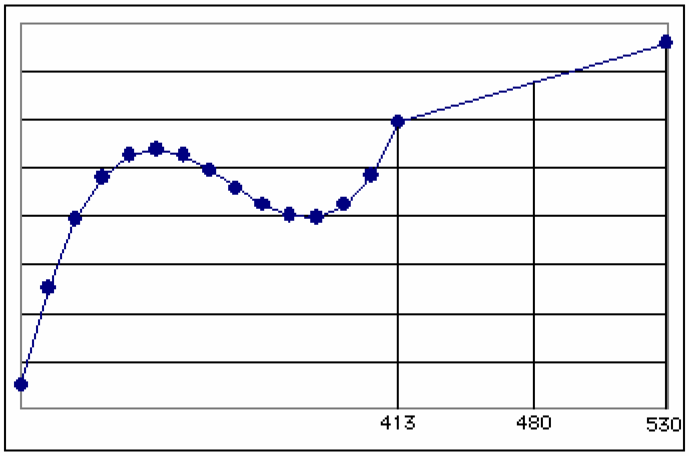
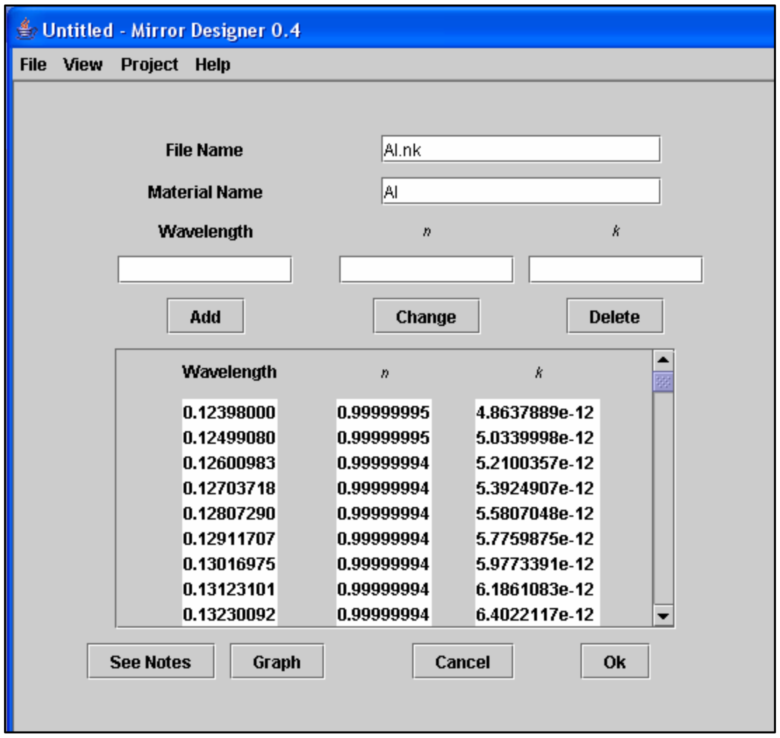


Figure 14: Interpolation problem

If you add data to a file it can cause potential problems with the Interpolation. Referring to the above example, say you add data to *Ac.nk* file at 530 Å. Then, some time later, you ask Mirror Designer to consider Actinium at some wavelength in between 413 Å and 530 Å, say 480 Å. Mirror Designer will interpolate to find this data and use it, but it will be very inaccurate because you did not give it any data near 480 Å.

3.3.5 Adding and Editing Materials

If you are using a local directory you can create new *.nk* files and even edit existing files using Mirror Designer. Click on the Add Material button to bring up the blank material window. Highlight a material and click the Edit Material button to bring up the material window and edit the material. If you click the Cancel button no changes are made to the file. Mirror Designer will warn you if you click the Ok button and



File Name: Al.nk
Material Name: Al

Wavelength	n	k
0.12398000	0.99999995	4.8637889e-12
0.12499080	0.99999995	5.0339998e-12
0.12600983	0.99999994	5.2100357e-12
0.12703718	0.99999994	5.3924907e-12
0.12807290	0.99999994	5.5807048e-12
0.12911707	0.99999994	5.7759875e-12
0.13016975	0.99999994	5.9773391e-12
0.13123101	0.99999994	6.1861083e-12
0.13230092	0.99999994	6.4022117e-12

Figure 15: Material window

are about to write over a file.

File Name - The file name must have a *.nk* extension. Mirror Designer will add it if you do not.

Add - You can add data to this file. Enter the wavelength and the *n* and κ values you want to add. Then click the Add button. Mirror Designer will warn you if there is already data for the wavelength.

Change - You can change data that already exists. Select a row of data and it will appear in the Wavelength, *n*, and κ fields. Change the *n* and κ data to the new values. Then click the Change button. You cannot change the wavelength using the change button.

See Notes - This button will display the notes for this file. You can modify the notes then return to the new material window.

Graph - This button will display a graph of the data in this file.

3.4 Advanced

The Advanced Panel allows you to modify important variables. Changing these variables can significantly change Mirror Designer's speed and accuracy.

Mirror Designer will start with reasonable values for each variable. At any time you may click the Reset to defaults button to return all the variables to the reasonable values. However, you may want to change the variables for various reasons.

1. You may want to be absolutely sure that you have the best possible mirror. You can change the variables so that Mirror Designer will run much longer, but will give a more exact answer.
2. You plan on using the local optimizer (See **2.1 Local vs. Global Optimization** and **4.4 Local Optimizer**).

3. You may need a quick answer, even if it isn't the best possible solution.
4. You may want to control how Mirror Designer generates new mirrors.

To understand how these variables influence how Mirror Designer performs you must have a basic understanding of the genetic algorithm that drives the program (see **2.2 The Genetic Algorithm**).

Population Size - The population size is the number of mirrors that Mirror Designer has in each generation. The population size also defines how much of the solution space the genetic algorithm will sample in its initial, random, generation. With the population size set too small, the genetic algorithm may not sample enough of the solution space to ever find the global maximum. With the population size set too high the genetic algorithm becomes very time intensive.

Determining a good population size depends on the size of the solution space, which is affected by several factors: the number of layers in the mirrors, number of materials used in the mirrors, the number of materials available, and the range of thicknesses allowed. The number of dimensions in the solution space is the number of materials in each mirror plus the number of layers in each mirror. The range of each dimension is either the number of materials available, or the range of thicknesses available. Since changing one material in a mirror will usually change the reflectance much more than changing the thickness of one layer, you should choose a population size large enough so that the initial population will contain several mirrors of each possible material combination. The number of possible material combinations is given by

$$\left(M_{Availbable}\right)^{M_{Mirror}}, \tag{23}$$

where $M_{Available}$ is the number of materials available, and M_{Mirror} is the number of materials in each mirror. Clearly, as more materials are used, the necessary population size increases very rapidly.

If you do not specify a population size, a default will be used. For aperiodic mirrors, Mirror Designer will set the default population size, $P_{Default}$, to

$$P_{Default} = \frac{N_{Layers}}{2} \left[\left(M_{Available} \right)^{M_{Mirror}} \right], \tag{24}$$

where N_{Layers} is the number of layers in each mirror. Thus, the initial population will have, on average, one mirror of each possible material combination for every two layers in the mirror. For periodic mirrors, Mirror Designer will set the default population size to

$$P_{Default} = M_{Mirror} \left[\left(M_{Available} \right)^{M_{Mirror}} \right]. \tag{25}$$

The initial population will have an average of one mirror of each possible material combination for each repeating layer.

<u>Epsilon</u> - The genetic algorithm will continue running until there is no significant improvement. Epsilon defines what is considered significant improvement. If Epsilon is 0.01, then as long as the merit value of the top mirrors changes by an average of at least 0.01 there is significant improvement. Lowering Epsilon will cause Mirror Designer to run longer, but find a more exact answer. If you plan on using the local optimizer (See **4.4 Local Optimizer**) you can set Epsilon to a higher value, such as 0.1 or higher.

<u>Num NoChanges</u> - This variable defines the number of consecutive generations that must occur without significant change before Mirror Designer will finish running. This prevents Mirror Designer from quitting if for some reason there is no significant change in one generation, but there would have been in the next. Generally two or three NoChanges are sufficient.

Num Mirror Avg - This variable defines the number of mirrors to consider when determining whether or not significant change has occurred.

Children Replace % - This variable defines the percentage of the population that will be replaced by children mirrors each generation. If Children Replace % is set to 70%, during each generation 70% of the population will be thrown away, and replaced by children mirrors.

Pick Parents From % - For each new generation the top mirrors are blended to make children mirrors. This variable defines the percentage of the mirrors that will be used to generate children mirrors. If it is set to 25%, only the top 25% of the mirrors will be used to create the new mirrors for the next generation. You should not pick parent mirrors from those mirrors being replaced. To avoid this, the sum of Child Replace % and Pick Parents From % should not be greater than 100%.

Mutant Replace % - The percentage of mirrors that will be mutated in each generation.

Pick Mutants From % - Defines the percentage of mirrors that will be considered for mutations.

Diversity - As Mirror Designer runs good mirrors tend to multiply and overrun the population. If this happens too soon, a mirror that may initially seem good can push out a mirror that would have evolved into an even better mirror. This variable allows Mirror Designer to avoid duplicate mirrors in the generation. The higher you set the diversity the more Mirror Designer will penalize mirrors that repeat materials.

3.5 Output

In the output panel you identify the file where Mirror Designer will send the output generated by the genetic algorithm. You can also specify the level of output you want; there are four settings: terse, moderate, extensive, and exhaustive.

Terse - This setting generates the least amount of output. It includes only the top 5 mirrors of the final generation.

Moderate - This setting includes the data for the included materials at wavelengths of interest. It also includes the top 5 mirrors from each generation and all of the mirrors in the final generation.

Extensive - This setting includes the data for the included materials and all the mirrors from each generation, including the final generation.

Exhaustive - This setting generates the most output. It includes all the output from the extensive setting as well as records of all changes to mirrors through both crossovers and mutations. This setting is useful when analyzing how crossovers and mutations are performed and is not commonly used.

4. MIRRORS SECTION

The Mirrors Section of Mirror Designer has four operations: general tolerance testing, tolerance testing targeting specific layers, graphing, and local optimization. There are four buttons, one corresponding to each operation. The Back button will take you back to the Project Section, and erase the current list of mirrors.

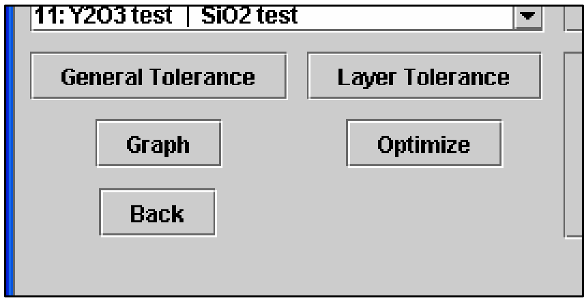


Figure 16: Mirror options

4.1 General Tolerance

We are interested in the mirror's tolerance to small changes in the thickness of each layer. Mirror Designer may predict that a particular mirror satisfies the conditions very well. However, if the mirror fails to satisfy the conditions when the thicknesses are changed by even small amounts the mirror is not practical. Using current technology multilayered mirrors cannot be made with each layer within an angstrom of the recommended thickness. The general tolerance function tells you how much each layer can vary and still result in a good mirror.

Through the process of testing this function it was found that tolerances are generally quite reasonable. Due to the nature of the genetic algorithm mirrors with very low tolerance are not easily found. Mirrors with low tolerance are represented by a sharp peak in the solution space. Generally, the sharper the peak the less of the solution space it covers, and the genetic algorithm will have less of an opportunity to sample it. There may be exceptions, but in all tests currently performed this was found to be the case.

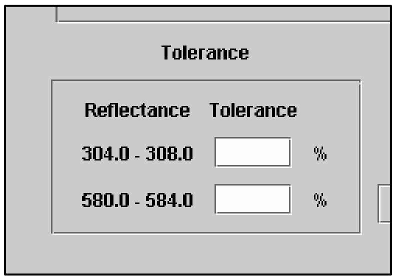


Figure 17: Tolerance

The General Tolerance button takes you to the General Tolerance Panel. The mirror being tested is shown at the top in a table. There are also two empty columns in the table where the range of acceptable thicknesses will be displayed.

Tolerance - You must specify the tolerance for each condition. Tolerance is measured in percentage reflectance. If you want to see you much each thickness can change while keeping each condition's reflectance within 1% of its original value, enter 1 for each condition's tolerance. There may be cases when you want one condition will be more tolerant to change than another.

Number of Steps - A large number of steps will run longer but yield a more accurate answer. For more information see **4.4.1 Process** below.

Step Size - A small step size will run longer but yield a more accurate answer. For more information see **4.4.1 Process** below.

4.1.1 Process

This section describes the process used to determine the mirror's tolerance to small changes in layer thickness. There are two parts to this process. Part one: the top layer's thickness is lowered until the mirror's reflectance changes past the specified tolerance. Then the thickness is raised until past tolerance. It is changed by the amount given by Step Size**.** This is repeated for each layer.

We are not interested in how much one layer can change while holding the others constant. So, in part two all the thicknesses are lowered simultaneously until the mirror's reflectance changes past the specified tolerance. Then the thicknesses are raised simultaneously until past tolerance. The amount the thickness is changed per step is different for the different layers. It is determined by the thicknesses calculated in part one and the Number of Steps. Each thickness is changed per step by the same percentage of the amount found in part one.

4.2 Layer Tolerance

The layer tolerance function allows you to target specific layers when testing the mirror's tolerance. Again you must specify a tolerance for each condition. (See **4.1 General Tolerance**) You must also select the two layers of the mirror to be compared. Enter the first layer number and the next layer will automatically be entered.

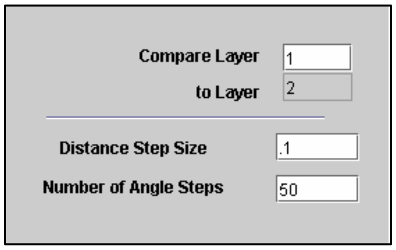

Compare Layer: 1
to Layer: 2
Distance Step Size: .1
Number of Angle Steps: 50

Figure 18: Layer tolerance options

Mirror Designer will display a graph plotting the thickness of one layer vs. the thickness of the other layer. The green center mark shows the thicknesses for these layers. The red ellipse shows the range of acceptable thicknesses to keep the mirror within the tolerance specified for each condition. The graph allows you to see how changes in the thickness of one layer affect the tolerance of another layer.

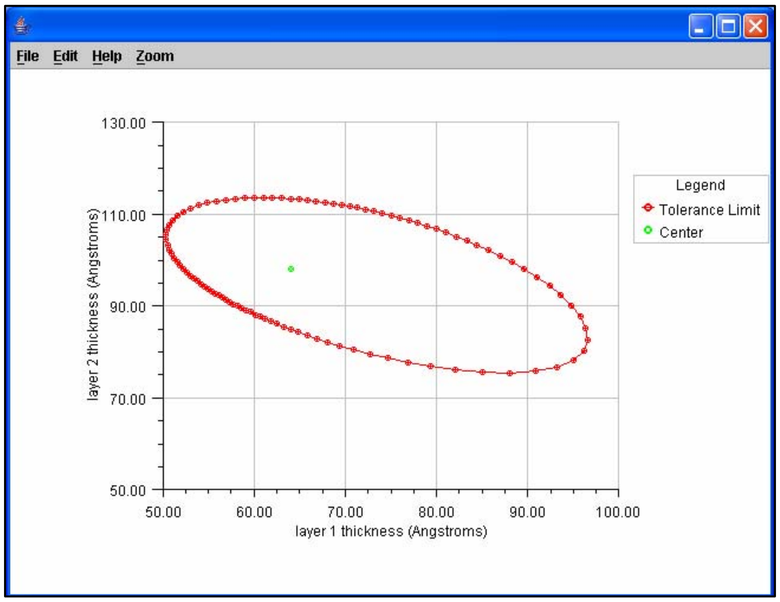


Figure 19: Layer tolerance results

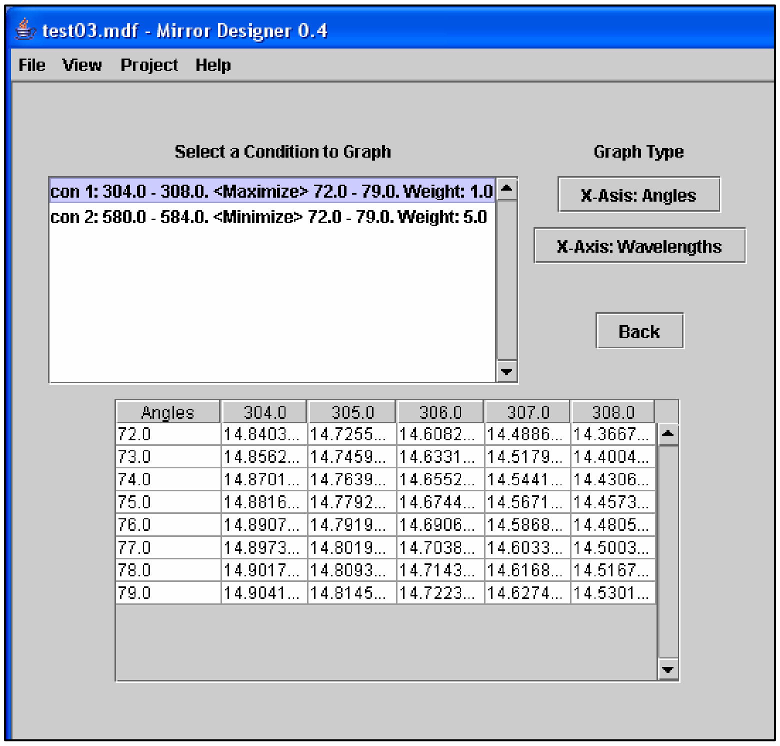
Number of Angle Steps - More angle steps will produce more points on the graph but require a longer run time.

Distance Step Size - Using a low step size produces more accurate points on the graph. If you enter 0.1, all points will be within 0.1 Å of the correct position.

4.3 Graph

When you highlight a mirror from the list, the mirror's attributes are shown to the right. The materials and thicknesses are displayed in the table on top, and the reflectances of the different conditions are displayed in the table below. The reflectances shown are the average reflectances across the range of wavelengths and angles in the condition. The graph function allows you to look closer at the reflectances at different angles and wavelengths.

You must select the condition you want to graph. When a condition is selected the data is displayed below. You can graph the reflectance with the angles along the X-Axis with the wavelengths forming different series of data, or with the wavelengths along the X-Axis with the angles forming different series of data.



con 1: 304.0 - 308.0. <Maximize> 72.0 - 79.0. Weight: 1.0
con 2: 580.0 - 584.0. <Minimize> 72.0 - 79.0. Weight: 5.0

Angles	304.0	305.0	306.0	307.0	308.0
72.0	14.8403...	14.7255...	14.6082...	14.4886...	14.3667...
73.0	14.8562...	14.7459...	14.6331...	14.5179...	14.4004...
74.0	14.8701...	14.7639...	14.6552...	14.5441...	14.4306...
75.0	14.8816...	14.7792...	14.6744...	14.5671...	14.4573...
76.0	14.8907...	14.7919...	14.6906...	14.5868...	14.4805...
77.0	14.8973...	14.8019...	14.7038...	14.6033...	14.5003...
78.0	14.9017...	14.8093...	14.7143...	14.6168...	14.5167...
79.0	14.9041...	14.8145...	14.7223...	14.6274...	14.5301...

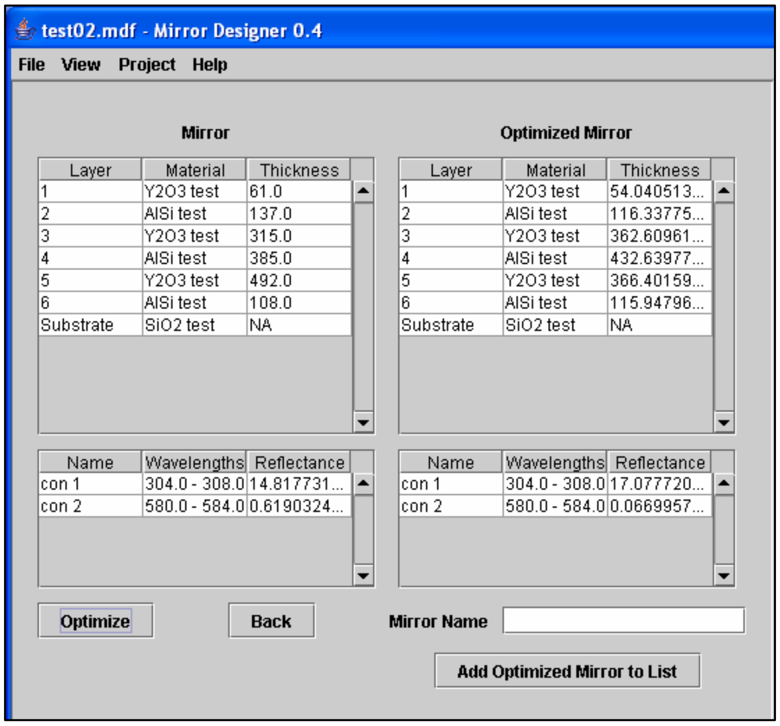
Figure 20: Graph window

4.4 Local Optimizer

The Optimize function performs changes on the thicknesses of the mirror to try and improve reflectivity. The optimize panel displays the mirror in the tables on the left. Click the Optimize button to perform the local optimization. The optimized mirror appears on the right. The new, optimized, mirror can be added to the list of mirrors. Enter a name for the mirror and click the Add Optimized Mirror to List, and you will see the new mirror now at the top of the list of mirrors.

The quickest way to find the optimal mirror is to use a combination of the global and local optimizers. You can run the genetic algorithm for a shorter time by using a higher value for Epsilon (See Epsilon in **3.4 Advanced**). The genetic algorithm supplies you with places to start the local optimizer, which will quickly take you to the nearest peak. You will then have the global maximum, as long as the genetic algorithm ran long enough that one of the nearest peaks is the global maximum.

Mirror Designer uses JMSL's MinConGenLin class to perform the local optimization.


test02.mdf - Mirror Designer 0.4
File
View
Project
Help
Optimize
Back
Mirror Name
Add Optimized Mirror to List

Mirror

Layer	Material	Thickness
1	Y2O3 test	61.0
2	AlSi test	137.0
3	Y2O3 test	315.0
4	AlSi test	385.0
5	Y2O3 test	492.0
6	AlSi test	108.0
Substrate	SiO2 test	NA

Name	Wavelengths	Reflectance
con 1	304.0 - 308.0	14.817731...
con 2	580.0 - 584.0	0.6190324...

Optimized Mirror

Layer	Material	Thickness
1	Y2O3 test	54.040513...
2	AlSi test	116.33775...
3	Y2O3 test	362.60961...
4	AlSi test	432.63977...
5	Y2O3 test	366.40159...
6	AlSi test	115.94796...
Substrate	SiO2 test	NA

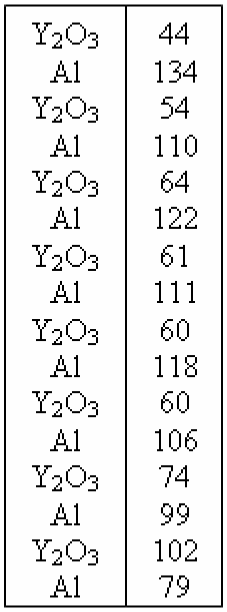
Name	Wavelengths	Reflectance
con 1	304.0 - 308.0	17.077720...
con 2	580.0 - 584.0	0.0669957...

Figure 21: Optimize window

5. CONCLUSION

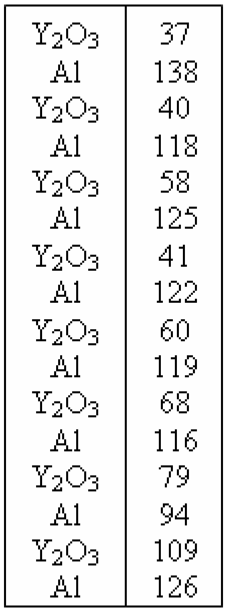
5.1 Testing

The results generated from Lunt's project were used to test Mirror Desginer. Lunt found that for mirrors with only four layers, the best mirrors for the IMAGE Mission problem contained U and Te. For mirrors with more than four layers, Al and Y_2O_3 were the best materials. When using the same optical constants used by Lunt's program the same results were reached. The thicknesses of all but two layers of the mirror found by Mirror Designer were within 15 Å of the thicknesses found by Lunt's program[6].



Y_2O_3	44
Al	134
Y_2O_3	54
Al	110
Y_2O_3	64
Al	122
Y_2O_3	61
Al	111
Y_2O_3	60
Al	118
Y_2O_3	60
Al	106
Y_2O_3	74
Al	99
Y_2O_3	102
Al	79

Figure 23: Mirror found by Lunt's program[6]
304 **Å** Reflectance: 35.12%
584 **Å** Reflectance: 0.19%



Y_2O_3	37
Al	138
Y_2O_3	40
Al	118
Y_2O_3	58
Al	125
Y_2O_3	41
Al	122
Y_2O_3	60
Al	119
Y_2O_3	68
Al	116
Y_2O_3	79
Al	94
Y_2O_3	109
Al	126

Figure 22: Mirror found by Mirror Designer
304 Å Reflectance: 35.68%
584 Å Reflectance: 0.08%

When testing the tolerance of the mirror it was discovered that the thicknesses of the lower layers could be changed dramatically without significantly affecting the reflectance. By the time the light penetrates through to these bottom layers most of the light has been reflected or absorbed. This explains the large difference in thicknesses of the bottom layer of the two mirrors. Essentially you could remove the bottom layer without affecting the mirror. If the bottom layer is removed from the mirror given in Figure 23, the reflectance only changes to 35.66% at 304 Å and 0.08% at 584 Å.

It is interesting to note that when using the optical constants provided by the IMD database, instead of the constants used by Lunt, the results were not entirely consistent with Lunt's conclusions. This demonstrates the sensitivity to even small changes in the optical constants, and the importance of having accurate data.

5.2 Further work

5.2.1 Speed

As the population size increases speed becomes an important issue. Although speeding up the algorithm has not yet been explored, there are several ways to potentially speed up the program. It is not uncommon for a program to spend 90% of its time in only 10% of the code. This is most certainly the case in Mirror Designer. When the genetic algorithm runs the program must calculate the reflectance of thousands of mirrors. Whenever the mirrors are altered, the reflectance must be calculated again. Each time the reflectance must be calculated at each wavelength and angle of interest. These calculations are performed by only a few short methods. If the code in these methods can be optimized to run even slightly faster each time, it will make a significant difference to the process as a whole.

Perhaps one way to significantly optimize the code is to eliminate the Complex class. The calculations require the use of many complex numbers; however, Java has no built in complex type. Each time a complex number is needed a new object of type Complex must be made. This time consuming process can be avoided by removing the Complex class and rewriting the code using two Doubles to represent each complex number: one Double for the real component and another Double for the imaginary component. It will be more difficult to write code to perform complex arithmetic in this fashion, and it will be more difficult for someone to read and understand the code. However, the code should run faster.

Another option to speed up the program is to parallelize it. A parallelized program is written so that it can run using multiple processors simultaneously. This is the basis of supercomputing. Parallelization works best if the lengthy process can be divided into many pieces that can all run independent of each other. If each part of the process has to wait for the previous part to finish before starting, having several processors available will not speed up the procedure. The process of calculating the reflectance of several thousand mirrors is ideal for parallelization. The list of mirrors can be divided between the processors, and because the reflectance of one mirror does not affect the reflectance of the others, the calculations can be performed simultaneously.

5.2.2 Roughness

Mirror Designer calculates the reflectance of each mirror assuming that each interface between layers is planar. When a multilayer mirror is made, this in not the case; each layer has some degree of roughness. The roughness affects the angles of reflection and transmission. The more roughness at each interface, the more the actual reflectance of a mirror will differ from the calculated value. Once the affects of roughness are better understood a feature may be added to

Mirror Designer allowing it to test how different levels of roughness change the reflectance of the mirrors.

5.2.3 Further Expansion

Mirror Designer can solve many types of problems. However, it could be expanded further. Work is currently being done to consider transmitted as well as reflected light. Also, the user may want to optimize variable not considered by the current genetic algorithm. For example, the user may want Mirror Designer to find not only the best mirror, but also the best angle of incidence to use the mirror at. As Mirror Designer is used, more types of problems will surely arise. Although the code is lengthy, I have tried to make it readable, and well documented so that it can be modified in the future.

References

[1] S. Lunt, “The Use of Genetic Algorithms in Multilayer Mirror Optimization,” BYU, 1999, pp. 1-38.

[2] J. Peatross. M. Ware, “Physics of Light and Optics,” BYU, 2004, pp. 33-41, 53-58.

[3] V. G. Kohn, “On the theory of reflectivity by an x-ray multilayer mirror,” Phys. Stat. Sol. (b), 187, 1995, pp. 61-70.

[4] L. G. Parratt, “Surface Studies of Solids by Total Reflection of X-Rays,” Physical Review, 45, 1954, pp. 359-369.

[5] Visual Numerics, “JMSL Numerical Library Reference Manual,” an unpublished work by Visual Numerics, 2003, pp. 50-56, 135-202.

[6] S. Lunt, “Design of bifunctional XUV multilayer mirrors using a genetic algorithm,” Journal of X-Ray Science and Technology, 9, 2001, pp 1-11.