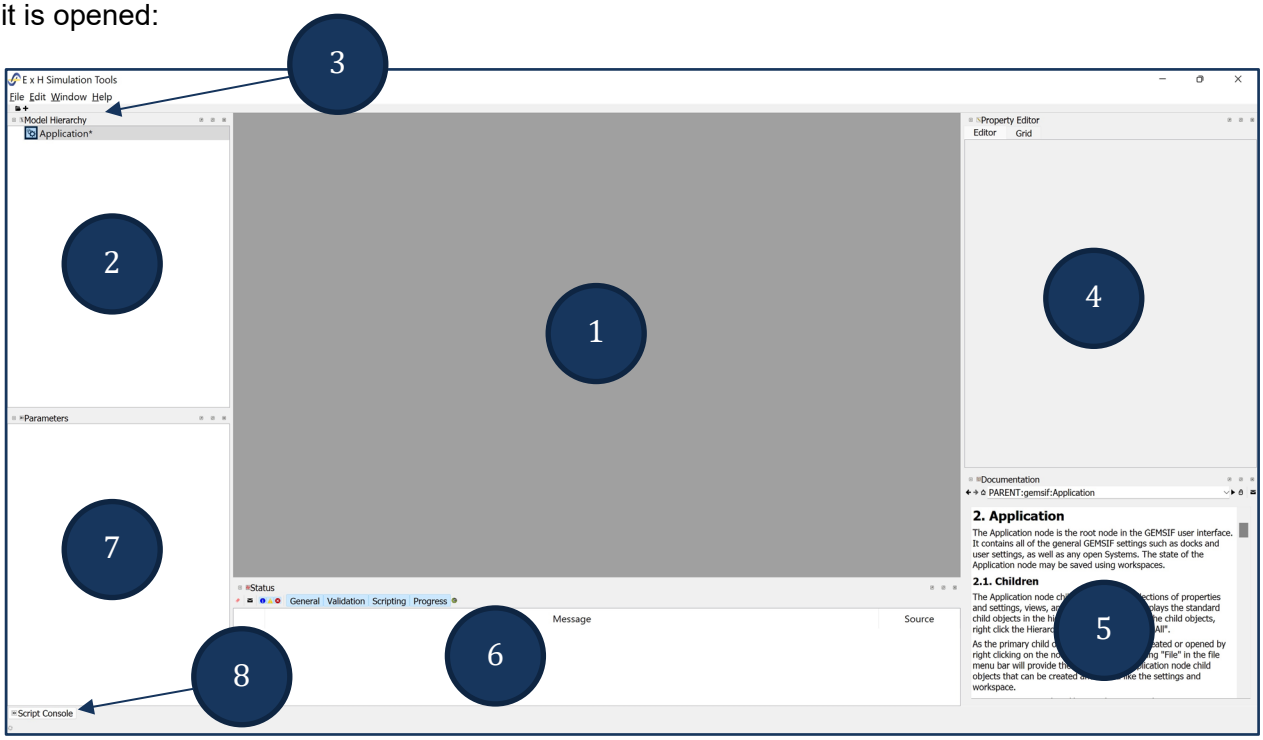
QUICK START GUIDE 1.0
Navigating reTORT/GEMSIF – Your Path to Optical Design
Estimated Time to Invest in Navigation portion pp 1 - 15: 20 minutes

By following this guide, you'll be able to easily navigate the reTORT Ray Tracer within our GEMSIF computational framework. We'll take you through entering a Tessar-style lens as a means to this end. Future guides will go further into designing, optimizing, and analyzing designs to help you continue to quickly gain familiarity with E x H software. You'll soon become an expert and be able to take advantage of all the advanced features reTORT has to offer.

One point to remember, GEMSIF is our common computational framework for all E x H solvers. Once in a while, as you continue to explore on your own, you'll come across features that are not common for optical systems – they are more than likely for PFSS, which while valuable for designing metasurfaces for use in reTORT, PFSS is also used for the same purpose in RF/Microwave systems. If you happen to include this in a reTORT analysis, in most cases, this will throw an error in the status box and the variable in question will likely turn red, letting you know to correct it.

First let's take a look at the major docks, all resizable at any time to suit your need, in the UI as it is opened:



1. The Workspace is where model views and results, plots and tabular data, appear. You can have multiple model views and an unlimited number of windows, split the windows or drag and place where you wish.
2. The Model Hierarchy dock is your command center. All you need to do can originate here. All that exists in your model is referenced when the hierarchy is expanded.

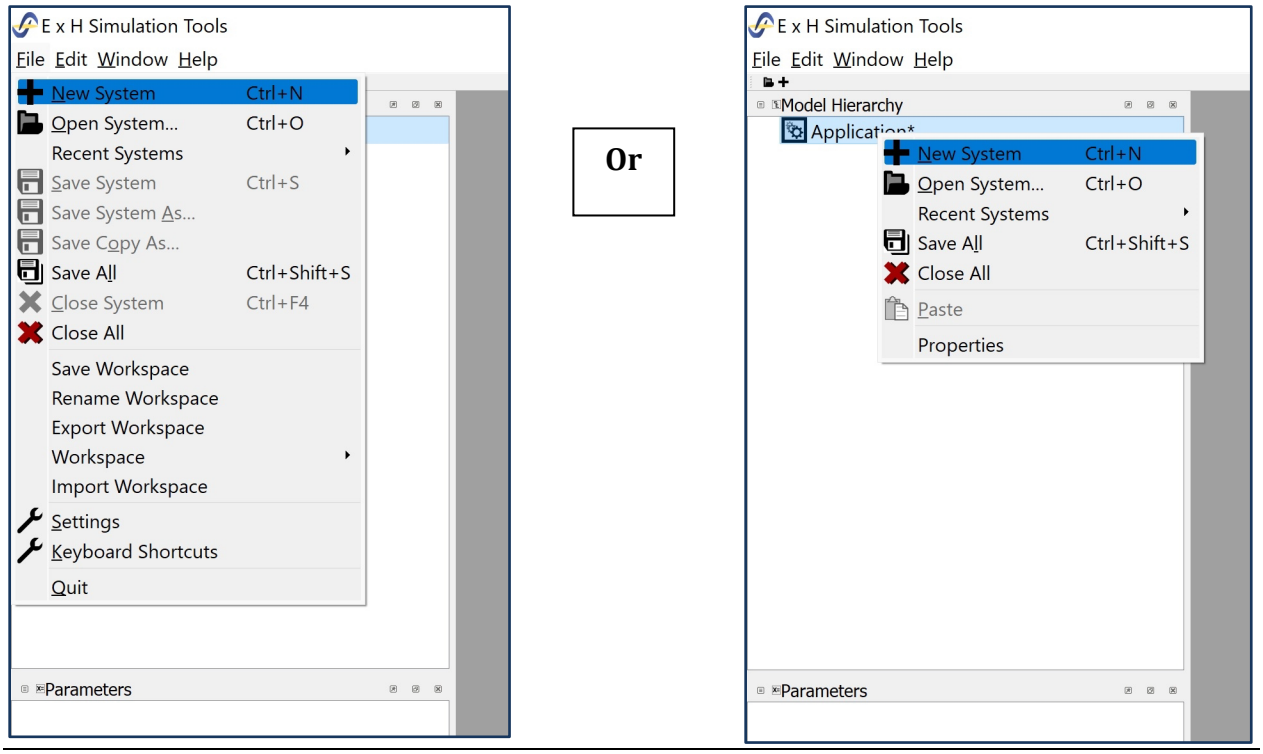
3. When you begin a new simulation, “quick command” Icons will appear below the main menu to run simulations or open the Optimization, Tolerancing and Plotting Wizards.
4. The Property Editor dock is your primary area for inputting data as well as specifying the types of outputs to be calculated. The applicable property editor will open automatically when you click an item in the model hierarchy.
5. The Documentation dock is the complete E x H contextual help. When you click an item in any dock with contextual help support, the documentation dock will instantly scroll to that subject in the User Manual.
6. The Status dock allows you to monitor progress but will also report errors in red, an invaluable dock to keep one eye on.
7. The Parameters dock is a one-stop shop for parameters used in common by many functions including the optimization and tolerancing wizards and result outputs.
8. This is a quick button for opening the Scripting dock adjacent to the Status dock, of course, you can move it anywhere.

These and other docks can also be opened in the main navigation from Windows>Add Docks.

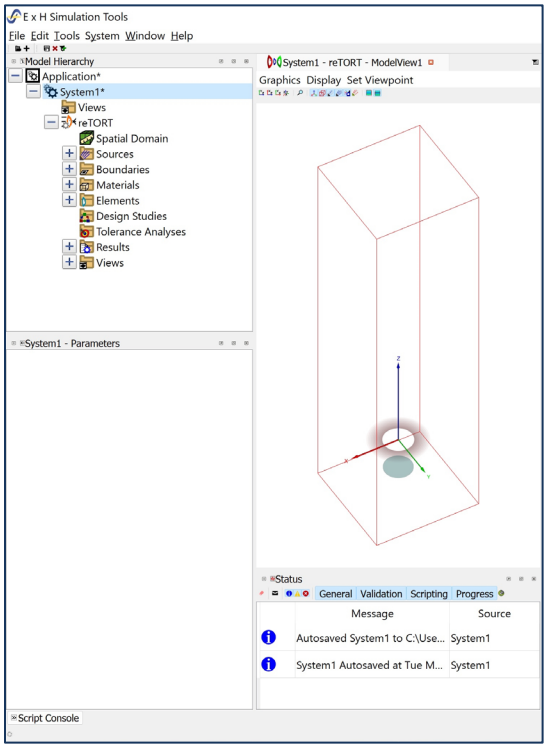
Now let’s learn to navigate all the basics by entering initial conditions for a simple Tessar-style lens that we’ll further design and analyze in future Quick Start guides.

Navigating by Setting up a Tessar-style Lens

First, initiate a new system from either File>New System or right click on Application in the model hierarchy and select New System. Notice in either case, you also have the option to open an existing system file or select among the last 10 systems you accessed:



Right click on the System1 you just created and click New Simulation>reTORT. Click on reTORT to expand the major headings of the model hierarchy to see:



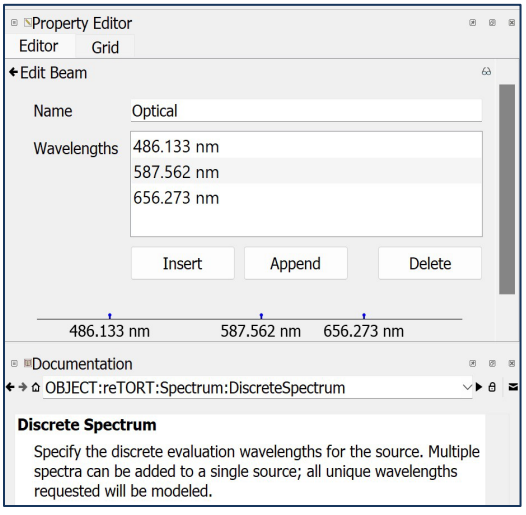
In reality, you're going to reach this stage where you're ready to start defining your design initial conditions and your objectives in about 15 seconds.

This is very simple stuff but we're creating a foundation from which we can move you through some of the most important relationships between docks.

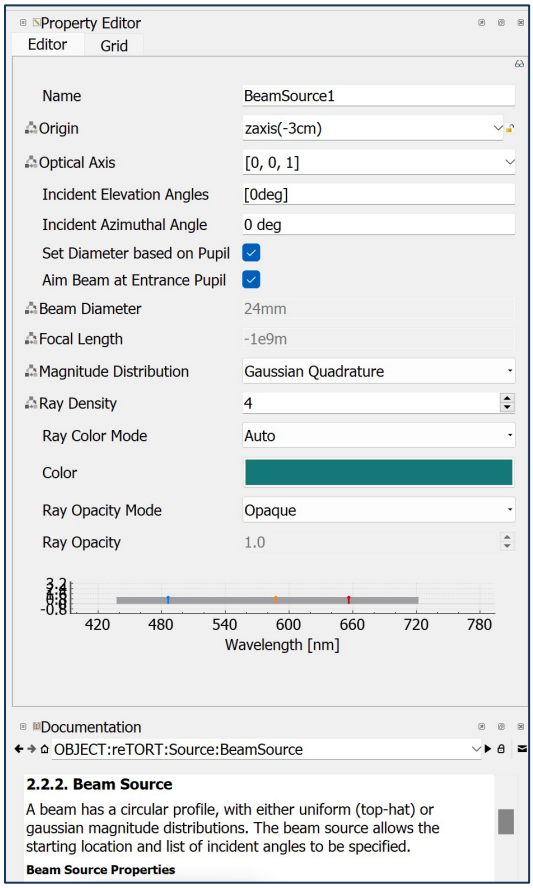
So, now let's proceed to the input.

Add Light Sources

Let's add our light source first. In the model hierarchy, expand Sources>BeamSource1>Spectral Lines>Optical and we'll see default wavelengths for analysis already populated. We're going to accept these default wavelengths for analyzing our design:



In the model hierarchy, click on Sources>BeamSource1 and in the Property Editor dock, you'll see the default data already populated and below, the documentation for a Beam Source:

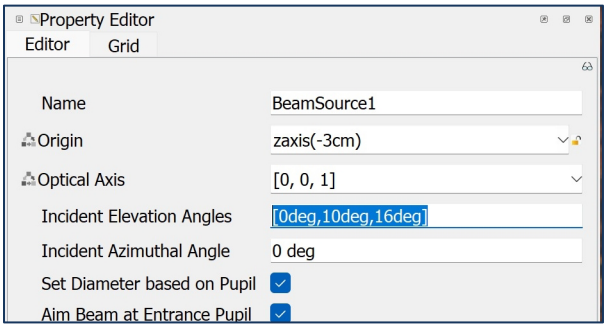


We're going to accept this default data as well, with the exception that we want to explore off-axis angles for this Tessar-style lens.

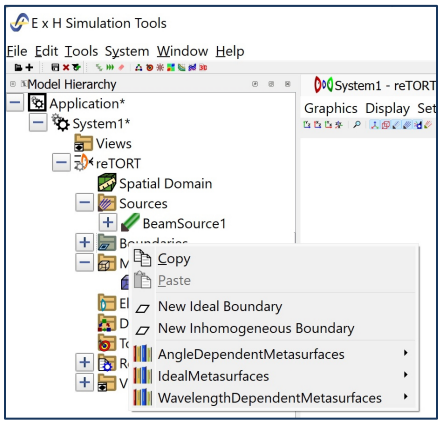
Note well that any of this data can be changed at any point in your analysis and new simulations and optimizations run.

In this case, it's handy to navigate under File and save candidate systems with a descriptive name, giving you the ability to either compare systems, or to come back to a prior system that behaved well, and further refine that one, abandoning other branches.

After adding additional angles to the Incident Elevation Angles property, that area of the property editor now appears as:



Boundaries

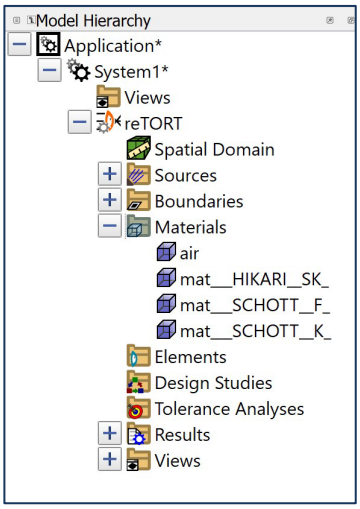


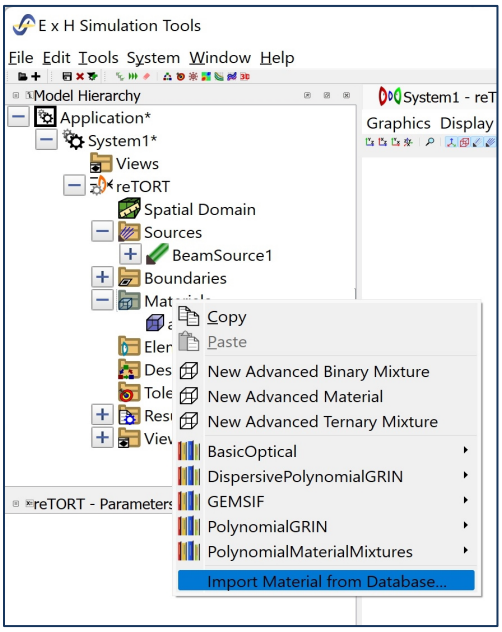
We will not use any special coatings or surfaces in this Tessar-style lens. But it's worthwhile noting the standard absorbing and perfectly reflecting definitions and the customizable options available when right clicking on Boundaries as shown here:

Now, Let's Add Materials We'll Use for this Lens

From the model hierarchy, right click on Materials>Import Material from Database

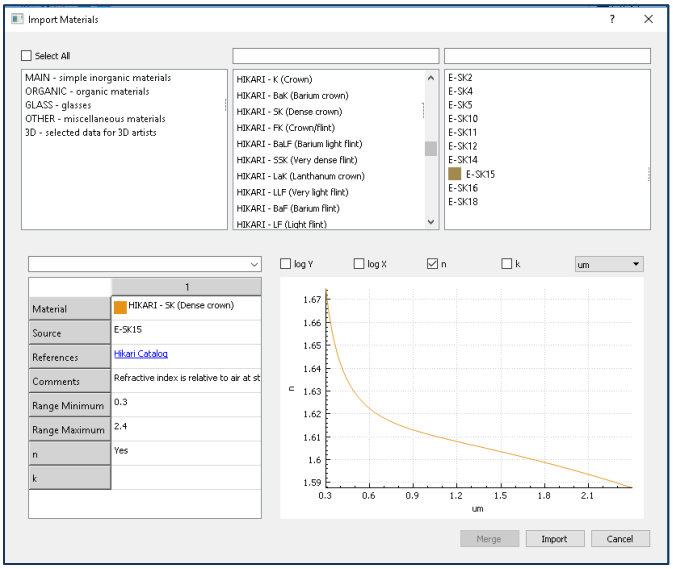
Note that in BasicOptical, you can define a new material with a specific refractive index.



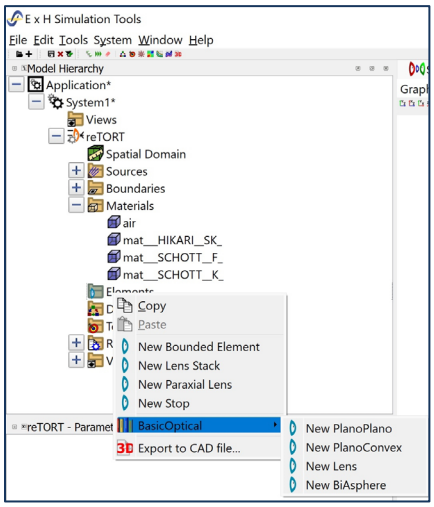


There are also definitions for a range of polynomial and dispersive polynomial GRIN profiles, and you can blend materials to create your own binary or ternary GRIN material.

We'll be treating GRINs, as well as the metasurface boundary options available in other quick start guides in the near future. In the Import Material window, we can scan for our choices but if we know the type of glass, or other material, we want to pull into our model, we can use the search function. Here we search and select Hikari E-SK15 dense crown glass:

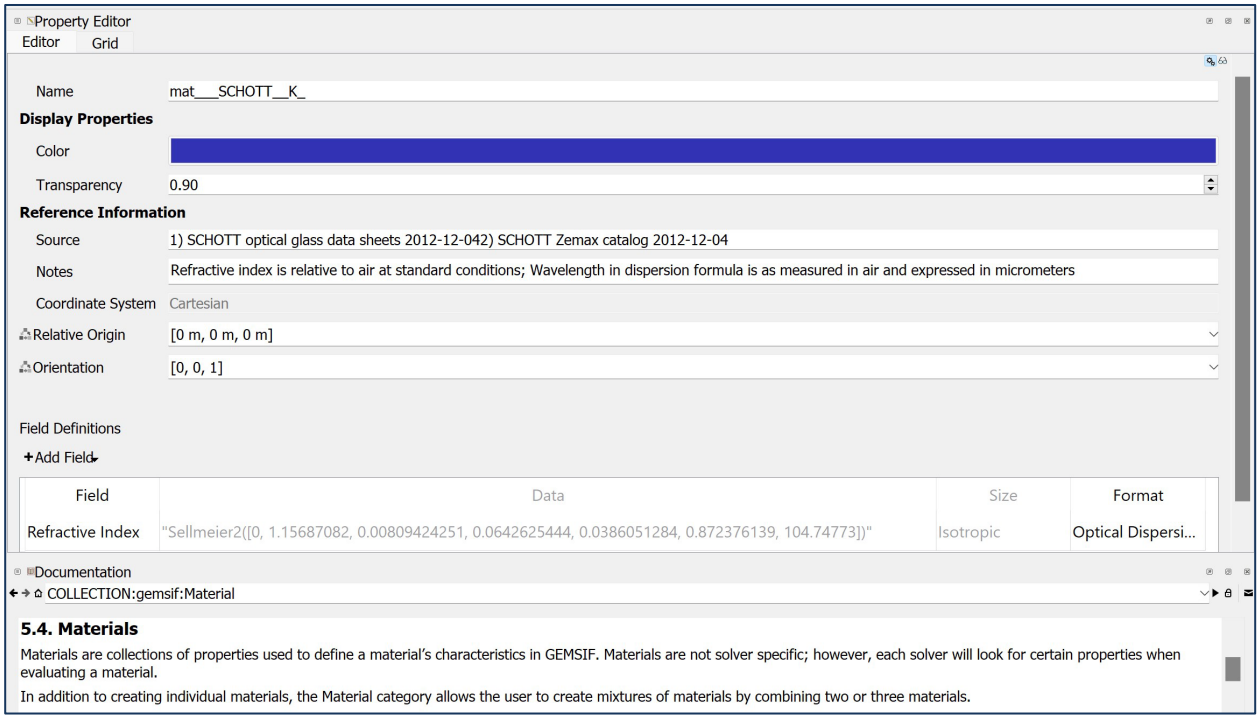


In the same manner, we will pull in Schott F2 and Schott K10, typical crown and flint glass materials. The Material list will now look like the following:



These are the materials we plan to use in this exercise. But we can add new materials at any time as we search for the best design to meet our objectives.

In the case of a standard catalog material, the property editor serves only as a repository of the material data, there is no input needed in this case.



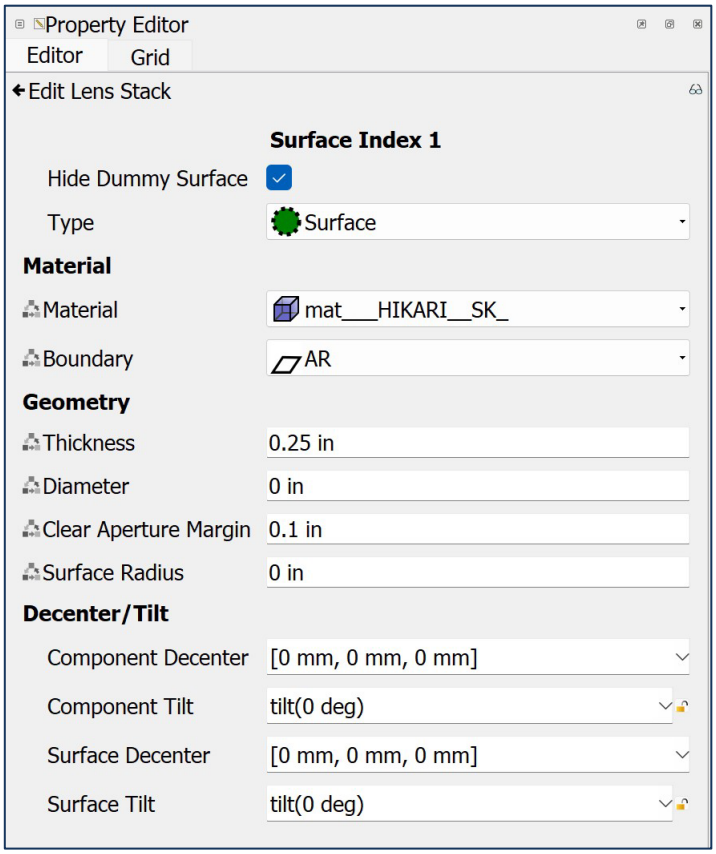
Now we're ready to add elements. We're going to start by adding the minimum number of elements. We can easily add or subtract elements, or change out one type of element for another, as our design/analysis proceeds.

Adding Elements – the basic lens stack

The Tessar-style lens is traditionally a sequential system. We plan to use reTORT's default non-sequential analysis regardless. Should we add elements or other features to reach design objectives, non-sequential ray tracing of even typical sequential system yields superior results

When we right click on Elements in our command center, the model hierarchy, we see options for bounded elements, lens stack, paraxial lens, addition of a stop, and several common initial shapes to begin our design work.

For this, we're going to select the Lens Stack, you'll find LensStack1 already initialized so open its Property Editor. We'll see that shortly but first, we're going to click on "Add" in the currently blank lens stack table, and the program will add the default, an air layer. We'll click edit and add our initial conditions for Surface1, which will look like this:



The Type is already "Surface", not image plane, stop or asphere, and we're not incorporating tilt/decenter in this design, so we'll only add the material and geometry data.

We can leave most data as zero since we're going to allow GEMSIF and its optimization wizard compute them. We can mix metric and English as GEMSIF will convert all to metric anyway.

Now we'll continue adding the rest of the lens stack starting from the source to the image plane. To keep this focused on how to use resource, we add thicknesses somewhat common but not exact for a Tessar-style lens. The final will look as below:

Surfaces

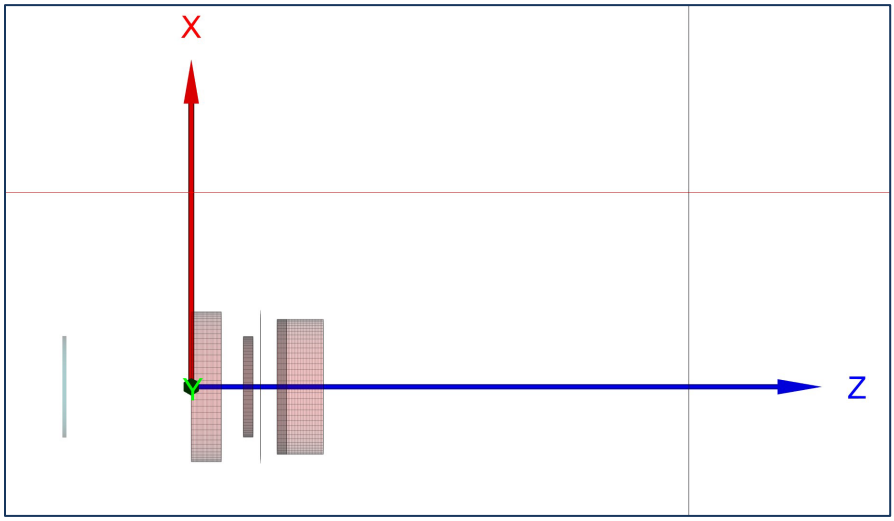
	Type	Material	Thickness	Radius	Diameter	Clear Aperture Margin
1	Surface	mat___HIKARI__SK_	0.2791 in	0 in	1.4 in	0 mm
2	Surface	air	0.2054 in	0 in	1.4 in	0 mm
3	Surface	mat___SCHOTT__F_	0.09 in	0 in	0.94 in	0 mm
4	Surface	air	0.0709 in	0 in	0.94 in	0 mm
5	Stop	air	0.1534 in	0 mm	0.715 in	0 mm
6	Surface	mat___SCHOTT__K_	0.09 in	0 in	1.26 in	0 mm
7	Surface	mat___HIKARI__SK_	0.3389 in	0 in	1.26 in	0 mm
8	Surface	air	3.4025 in	0 in	1.26 in	0 mm
9	Image Plane	air	---	0 mm	1e+15 m	0 mm

You'll notice that in creating our model view, the primary tool icons became visible between the main menu and the model hierarchy – mouse over to see the function of each but the center three

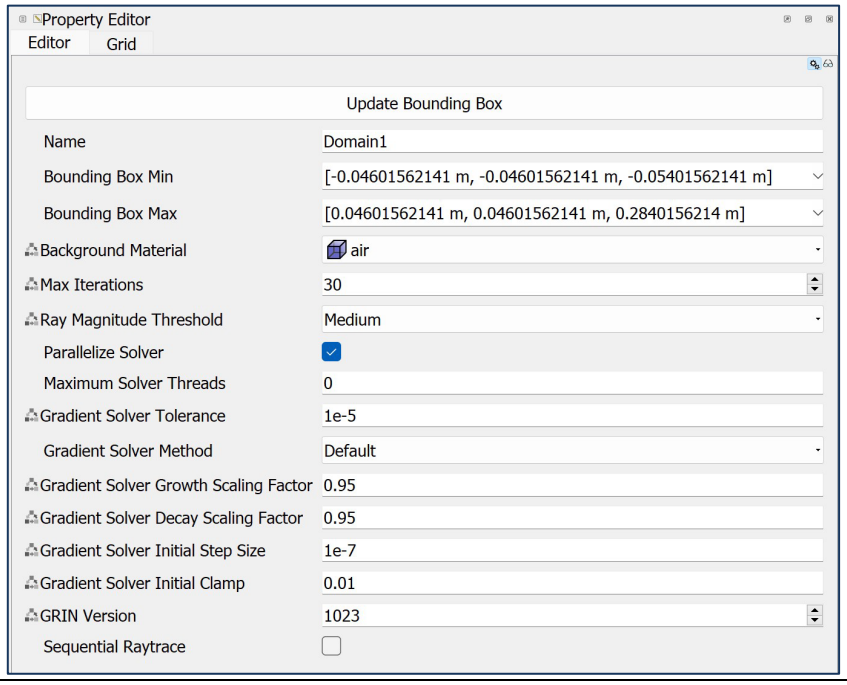
give one-click control over validation and running a simulation and the rightmost icons provide one-click control over the optimization, tolerancing and plotting wizards – please try them.

Below the model view menu, the model view icons have appeared:

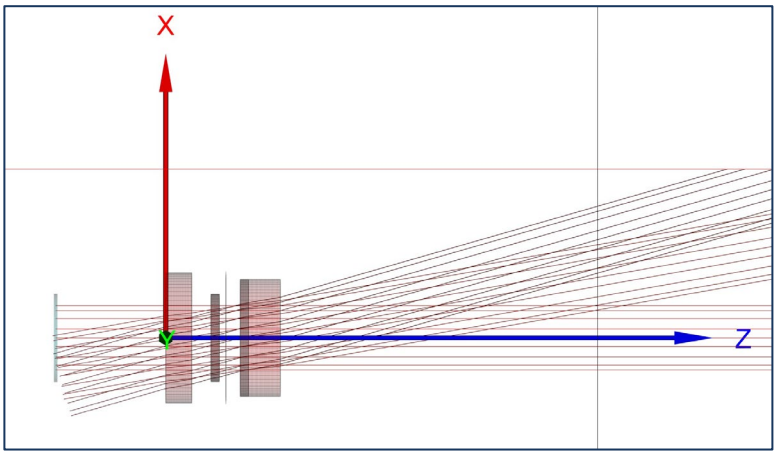
Again, mouse over to see the function of each. The first four provide instant realignment of the model view and we're going to click the 2nd one to align the model along the Y-axis with the source at the left:



Our bounding box was set to fit automatically but can also be manually set. The bounding box is the limit of our computational space and, if we shrunk the model view above slightly, we'd see it defined by red lines arranged in a hexahedron. Beyond that space, rays are no longer traced, all relevant objects of our design reside inside the bounding box. If you wish to alter this space, make sure the model hierarchy is expanded, click on Spatial Domain, and go to the far right to the spatial domain property editor, here we show the advanced screen:



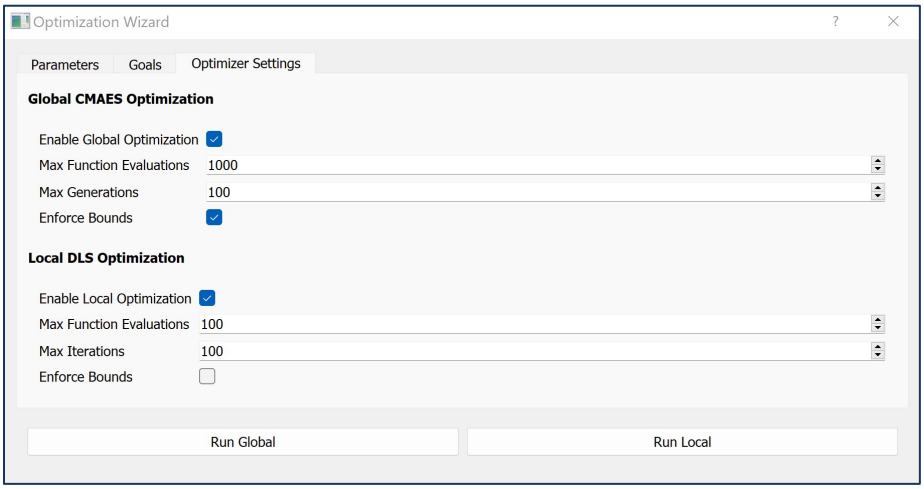
If we run a simulation, we'll see our full initial conditions for the model. In the middle of the icons, click Clear All Data, then click Validate All and note confirmation in the status dock, and then click Run All and the rays as they exist pre-design, pre-optimization will appear. You're now ready to begin optimization:



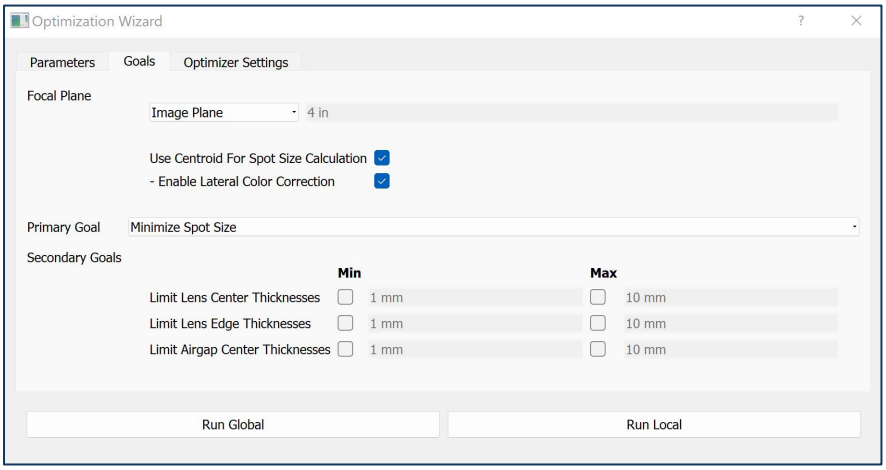
That Completes Initialization of your model - Design & analysis will be covered in Quick Start Guide 2.0 – But to complete the Navigation Topic, let's look at a few other functions

The Optimization and Tolerancing Wizards and Adding a Parameter

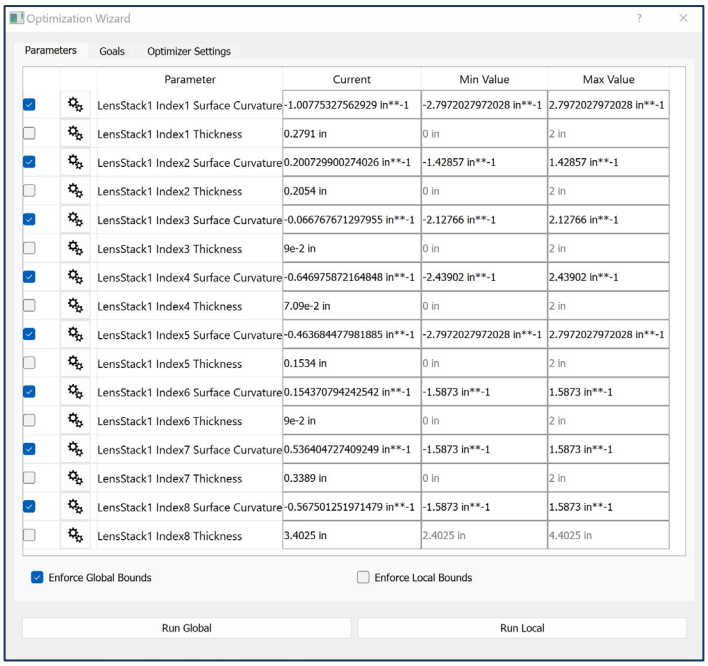
Open the Optimization Wizard. Most choices will be pre-loaded and you'll usually find most of these defaults to be acceptable for your design. First are the optimizer settings:



Next click Goals:



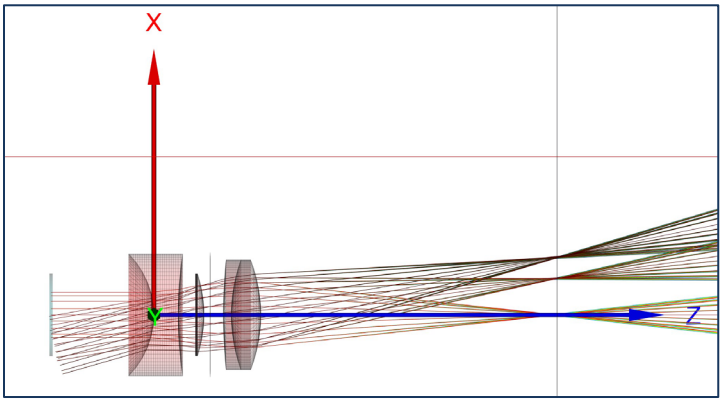
And in the third tab, Parameters, all have been pulled in automatically from the Lens Stack, Bounded Elements and other objects. Note that Min and Max values have been specified, feel free to review them, or run your optimization and inspect the results before changing them. You should usually find these values quite appropriate for your analysis:



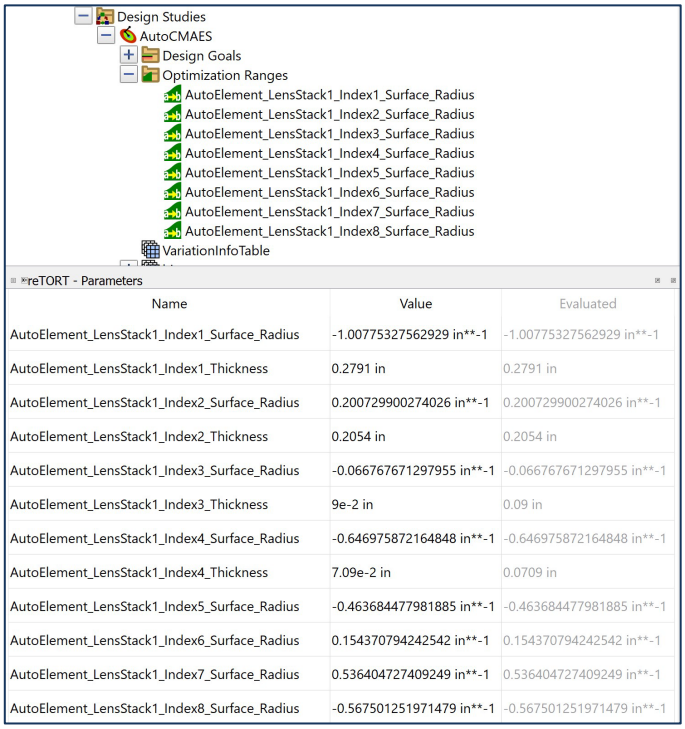
We've just checked every surface to be optimized and we're going to run a global optimization using CMAES. **Note: Optimizing every surface at once is neither good design practice nor a time saver. Guide 2.0 may take this in a different direction so this is just for exhibit here to illustrate the navigation in a short time. In fact, optimizing so many surfaces at once increases your chances of exceeding the bounds set and causing a needlessly long**

optimization run. Design is a more well thought out, pre-planned and iterative process. We'll make some short comments about design at the end.

Our optimization ran the maximum 1,000 simulations so we know we have more work to do. If we look at the model view, we can also see from the curvatures this is not a Tessar-style lens. We'll need to adjust parameters and goals in Guide 2.0 to design this to be a Tessar-style lens:

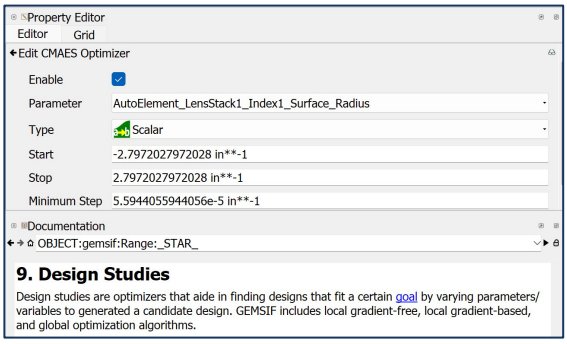


Note the parameters dock. In the value column, we can express any parameter as an equation and it will be evaluated and the current value displayed in the Evaluated column. If we wish to make one parameter a function of another variable, we can insert it in this one place and any function using that parameter will have that new evaluation of the parameter available for use at all times. We can also see the AutoElements created by the optimization wizard in the model hierarchy:

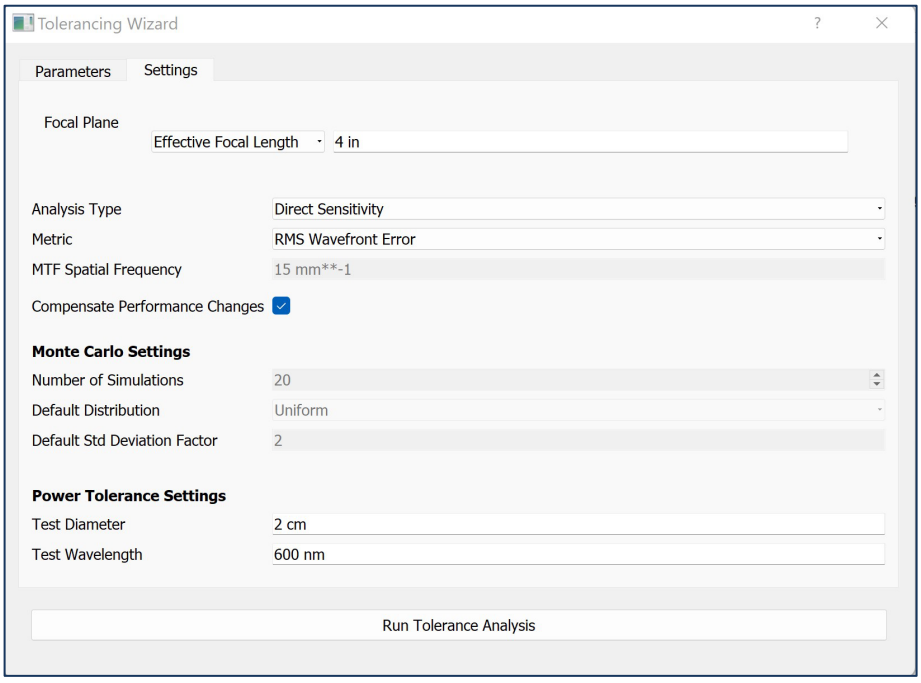


Name	Value	Evaluated
AutoElement_LensStack1_Index1_Surface_Radius	-1.00775327562929 in**-1	-1.00775327562929 in**-1
AutoElement_LensStack1_Index1_Thickness	0.2791 in	0.2791 in
AutoElement_LensStack1_Index2_Surface_Radius	0.200729900274026 in**-1	0.200729900274026 in**-1
AutoElement_LensStack1_Index2_Thickness	0.2054 in	0.2054 in
AutoElement_LensStack1_Index3_Surface_Radius	-0.066767671297955 in**-1	-0.066767671297955 in**-1
AutoElement_LensStack1_Index3_Thickness	9e-2 in	0.09 in
AutoElement_LensStack1_Index4_Surface_Radius	-0.646975872164848 in**-1	-0.646975872164848 in**-1
AutoElement_LensStack1_Index4_Thickness	7.09e-2 in	0.0709 in
AutoElement_LensStack1_Index5_Surface_Radius	-0.463684477981885 in**-1	-0.463684477981885 in**-1
AutoElement_LensStack1_Index6_Surface_Radius	0.154370794242542 in**-1	0.154370794242542 in**-1
AutoElement_LensStack1_Index7_Surface_Radius	0.536404727409249 in**-1	0.536404727409249 in**-1
AutoElement_LensStack1_Index8_Surface_Radius	-0.567501251971479 in**-1	-0.567501251971479 in**-1

In the model hierarchy, each parameter has been assigned as an AutoElement under optimization ranges and each can also be modified in its property editor by clicking the AutoElement as:

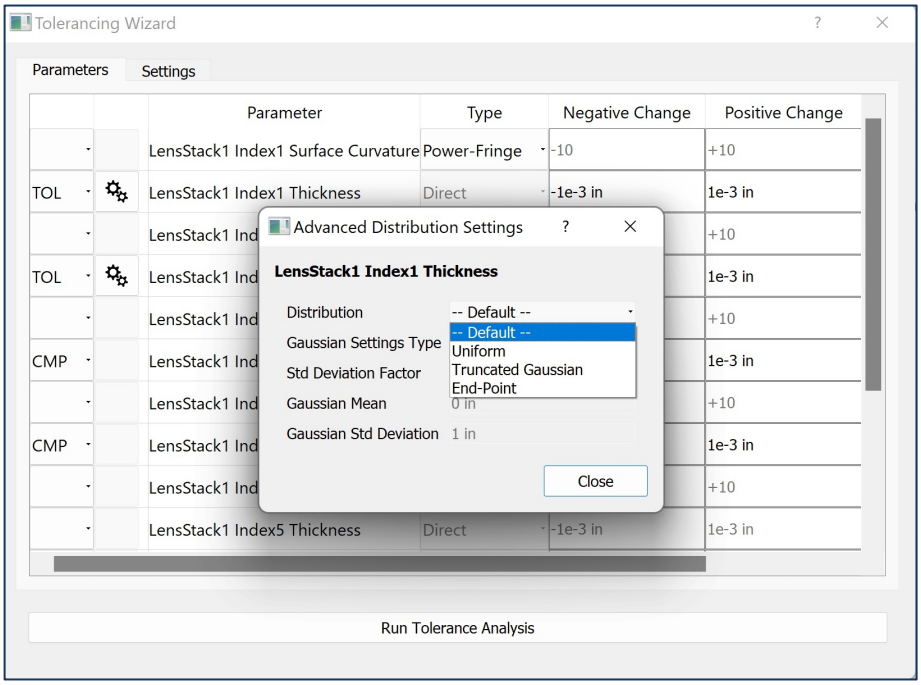


Before we demo adding a totally new parameter, let's open the tolerancing wizard. As soon as we did, the typical AutoTolerance parameters were estimated and populated in the parameters dock, which we'll take a look at soon. First, the tolerancing settings tab with default values estimated from the prior input and your choice of Analysis, direct sensitivity or Monte Carlo:



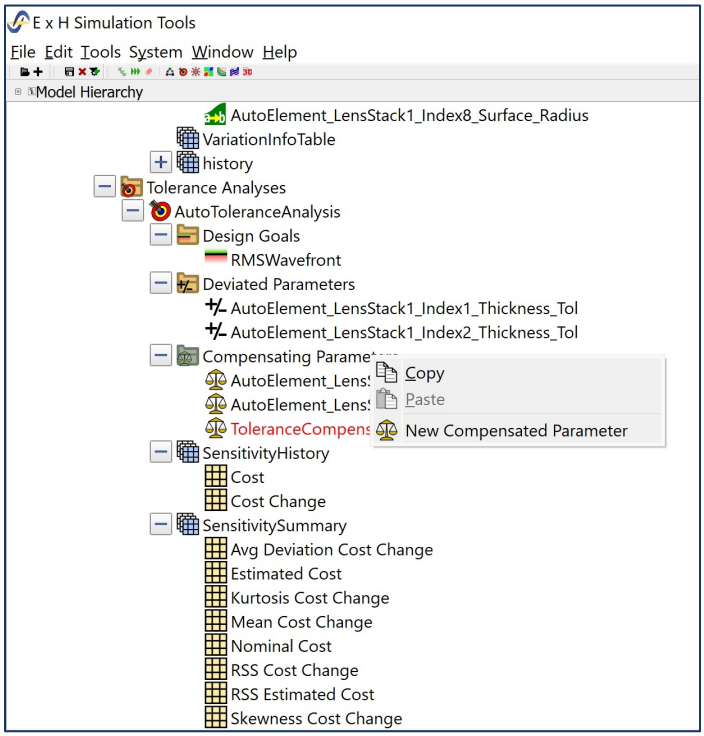
Next the tolerancing parameters tab where we can specify which parameters to deviate in either a sensitivity or Monte Carlo analysis and which we want to designate compensators to attempt to retain the design performance despite those deviations. Let's choose some thicknesses for deviation and compensation to show how they will also be added to our parameters dock for use by all program functions.

We've chosen these at random. Note also that you have a choice of distributions for deviations using the double gear icon:



Before we look at the parameters dock one last time, let's add a new parameter, just for this exercise, not necessarily of use in the later design and analysis. Let's right click in the center of the parameters dock and select New Parameter.

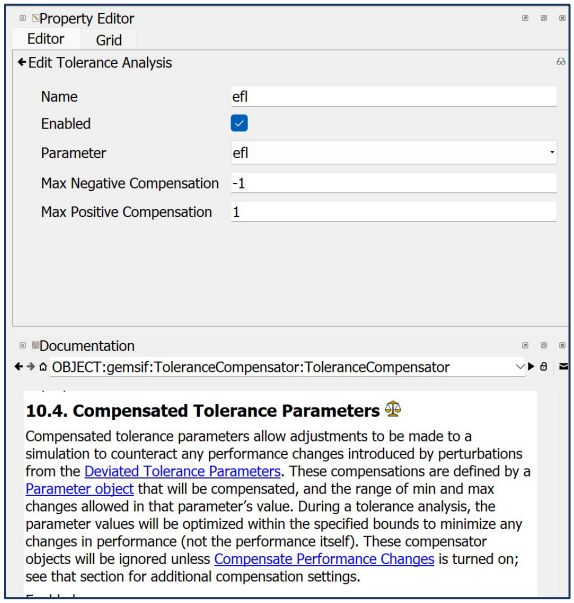
Parameter1 will appear in the dock. Let's double click on it and assign it the name efl which we'll be able to find in parameters elsewhere.



We want to use efl (lower case not to confuse the program with its known variable EFL) for compensation so in the model hierarchy, expand AutoToleranceAnalysis, right click on Compensating Parameters and click New Compensated Parameter.

ToleranceCompensator1 will appear.

Click on ToleranceCompensator1 and go to the property editor. For parameter, in the drop down box populated from the parameter dock, we select efl and we accept the default max deviations of -1 and +1:



With parameters pulled in from the optimization and tolerancing wizards and the one we added, the parameters dock appears as follows. This is a powerful tool, particularly with parameters calculated as a function of another independent property, for which the parameters dock can be used to rapidly change the value of that independent properties and populate it throughout the design study.

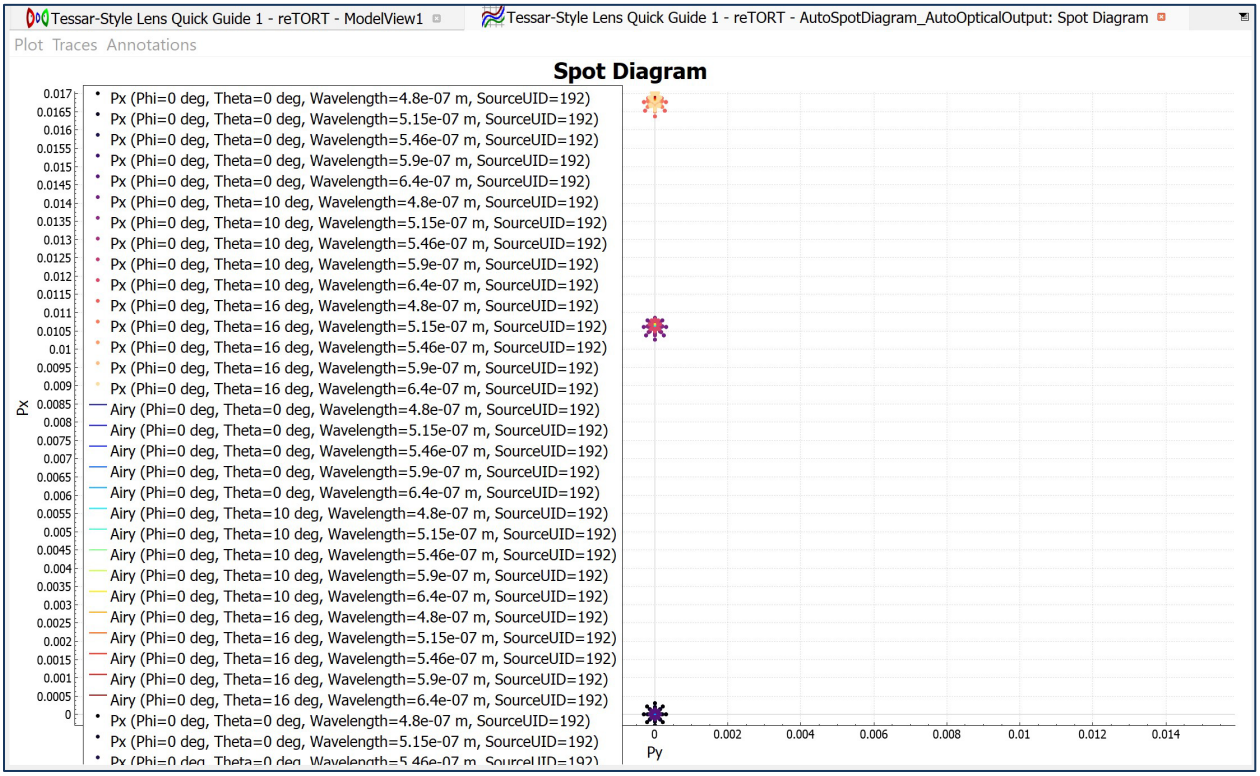
reTORT - Parameters

Name	Value	Evaluated
AutoElement_LensStack1_Index1_Surface_Radius	-0.388776648893745 in**-1	-0.388776648893745 in**-1
AutoElement_LensStack1_Index1_Thickness	0.2791 in	0.2791 in
AutoElement_LensStack1_Index2_Surface_Radius	-0.372601712225856 in**-1	-0.372601712225856 in**-1
AutoElement_LensStack1_Index2_Thickness	0.2054 in	0.2054 in
AutoElement_LensStack1_Index3_Surface_Radius	-1.00201227096834 in**-1	-1.00201227096834 in**-1
AutoElement_LensStack1_Index3_Thickness	9e-2 in	0.09 in
AutoElement_LensStack1_Index4_Surface_Radius	-0.839082111332711 in**-1	-0.839082111332711 in**-1
AutoElement_LensStack1_Index4_Thickness	7.09e-2 in	0.0709 in
AutoElement_LensStack1_Index5_Surface_Radius	-0.716306120146748 in**-1	-0.716306120146748 in**-1
AutoElement_LensStack1_Index6_Surface_Radius	-0.146940019084919 in**-1	-0.146940019084919 in**-1
AutoElement_LensStack1_Index7_Surface_Radius	0.0375467965560852 in**-1	0.0375467965560852 in**-1
AutoElement_LensStack1_Index8_Surface_Radius	-0.571939177750891 in**-1	-0.571939177750891 in**-1
AutoTolerance_Distribution	"Uniform"	Uniform
AutoTolerance_StdDeviationFactor	2	2
AutoTolerance_TestDiameter	2 cm	2 cm
AutoTolerance_TestWavelength	600 nm	600 nm
EFL	3 + 1	4

A Quick Look at the Spot Diagrams of our Example

For the sake of creating our navigation example, we took some broad steps which caused our overly-specified optimization analysis to go out of bounds many times, causing us to repeat the run. Still, taking a look at the spot diagrams, this example optimization, while not resulting in a Tessar-style lens, gave decent performance to use as a base for analysis and further design toward our final performance goals.

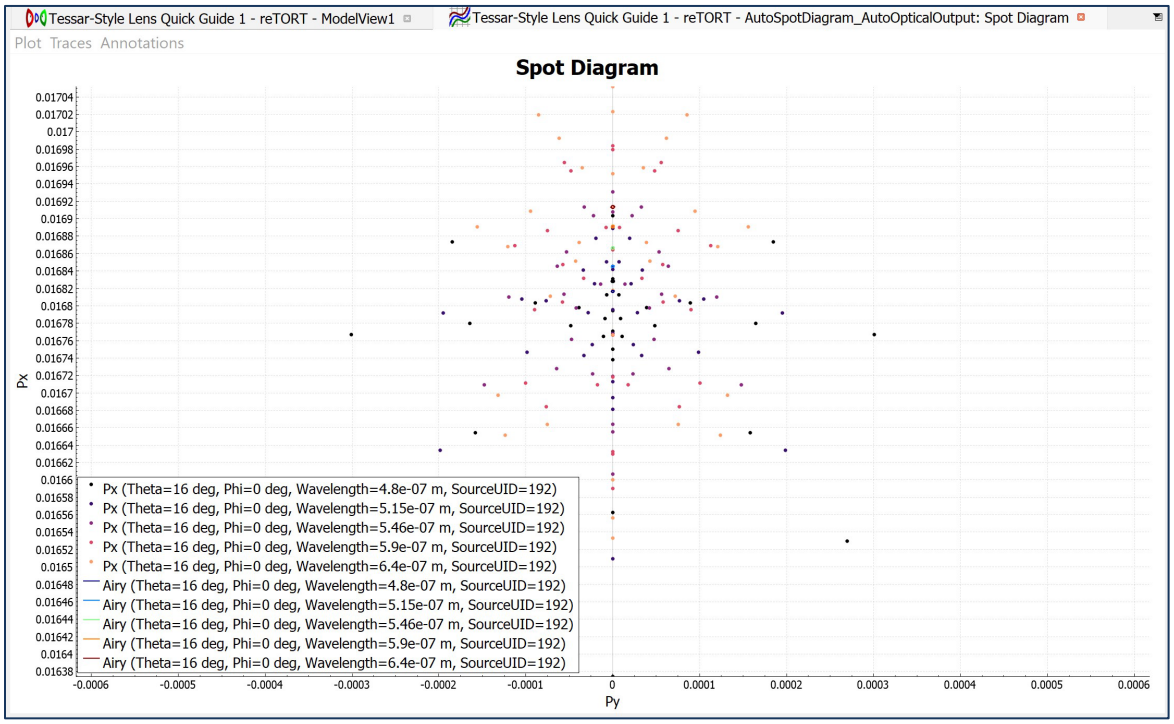
First, the spots from all three angles:



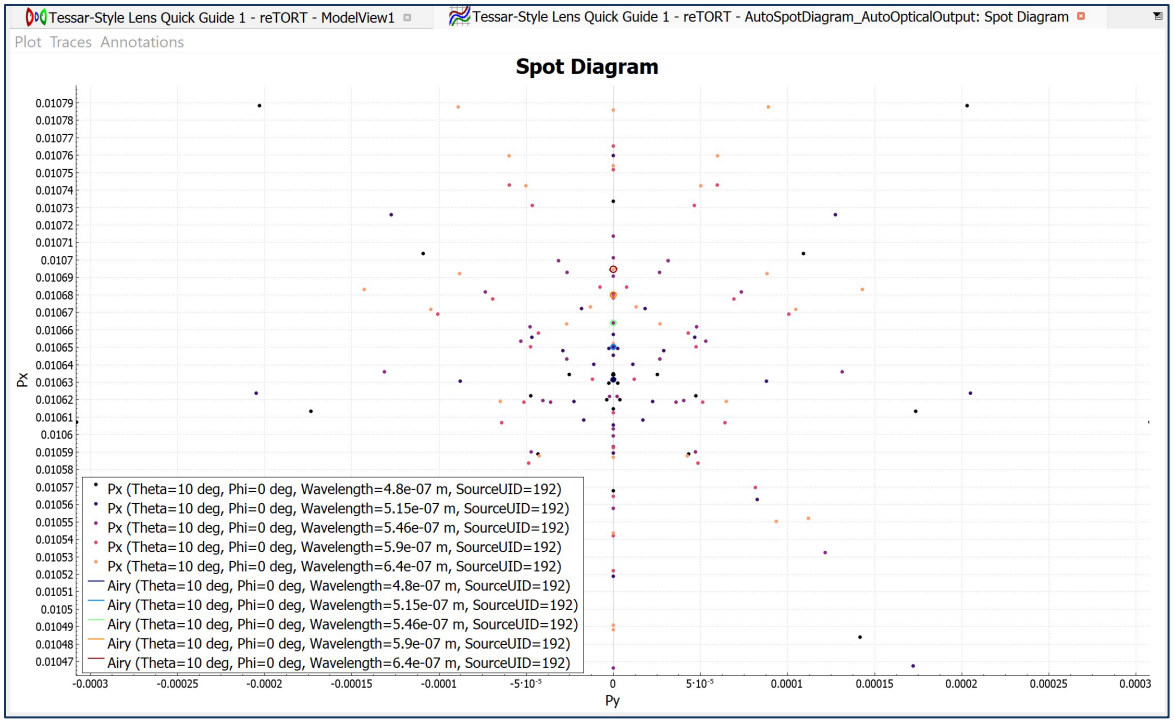
On a macro scale, the three spots are not so very different although there are indications of aberrations that need to be addressed at the off-axis angles.

Next, each of the three bundles:

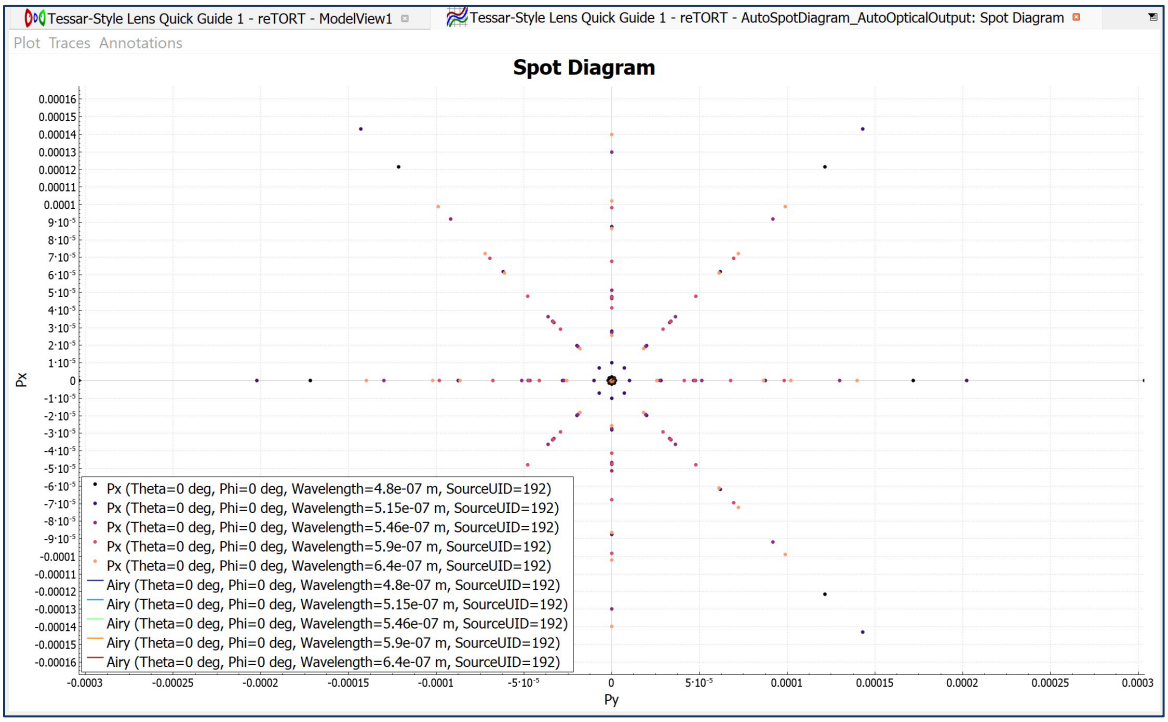
The 16 degree off-axis bundle at the image plane:



At 10 degrees off-axis:



And finally, the on-axis spot diagram - 0 degrees:



So, for one rough all surface optimization in which we, admittedly by error included the dummy spot surface, another factor that caused the optimization to sometimes hang-up, it turned into a pretty good start to the design.

Some notes about optical lens design

Designing an optical lens system is not a matter of setting up a ray tracer, inserting lenses, running an optimization and bingo, here's a lens system.

Design begins with performance and mechanical/dimensional requirements. Then, the designer needs to think through how to proceed with the design, even doing a few spreadsheet or graphical studies to frame the components the designer will use. Then optimization, while possible on a grand scale, is best done iteratively, but with the experience of the designer making wise choices of which components to optimize first, studying intermediate results at dummy surfaces, and then planning out next steps. Sometimes those steps are one or two back before making two or three forward. And then there's analysis, for example, to minimize aberrations to meet planned performance.

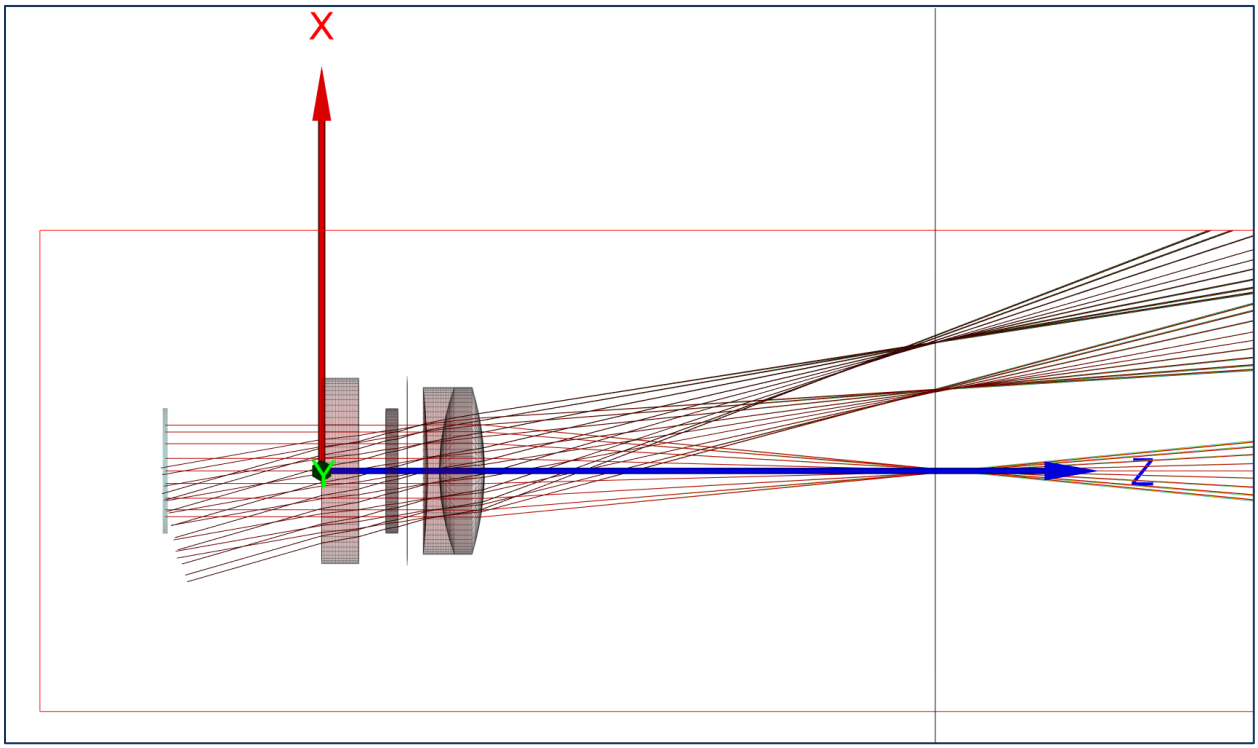
Along the way, as a means of working toward a manufacturable lens, we'll perform sensitivity analysis, sometimes even an intermediate Monte Carlo analysis. We'll always finalize the optical design with a full Monte Carlo analysis before judging the design viable for manufacturing.

That omits many steps. Optical lens design must be a results-driven, well thought out but iterative process that ultimately, after many steps, zeros in on a cost-effective, manufacturable product.

As an example, let's take a step-by-step approach to the example above, at least up to a point.

We will take advantage of the high speed of reTORT's CMAES global optimizer. When you use the iterative approach and do not try to optimize too many potentially contradictory parameters at one time, each optimization completes within seconds. Coupled with the typical optical designer's experience, reTORT's highly accurate global optimizer speeds the overall iterative approach to design.

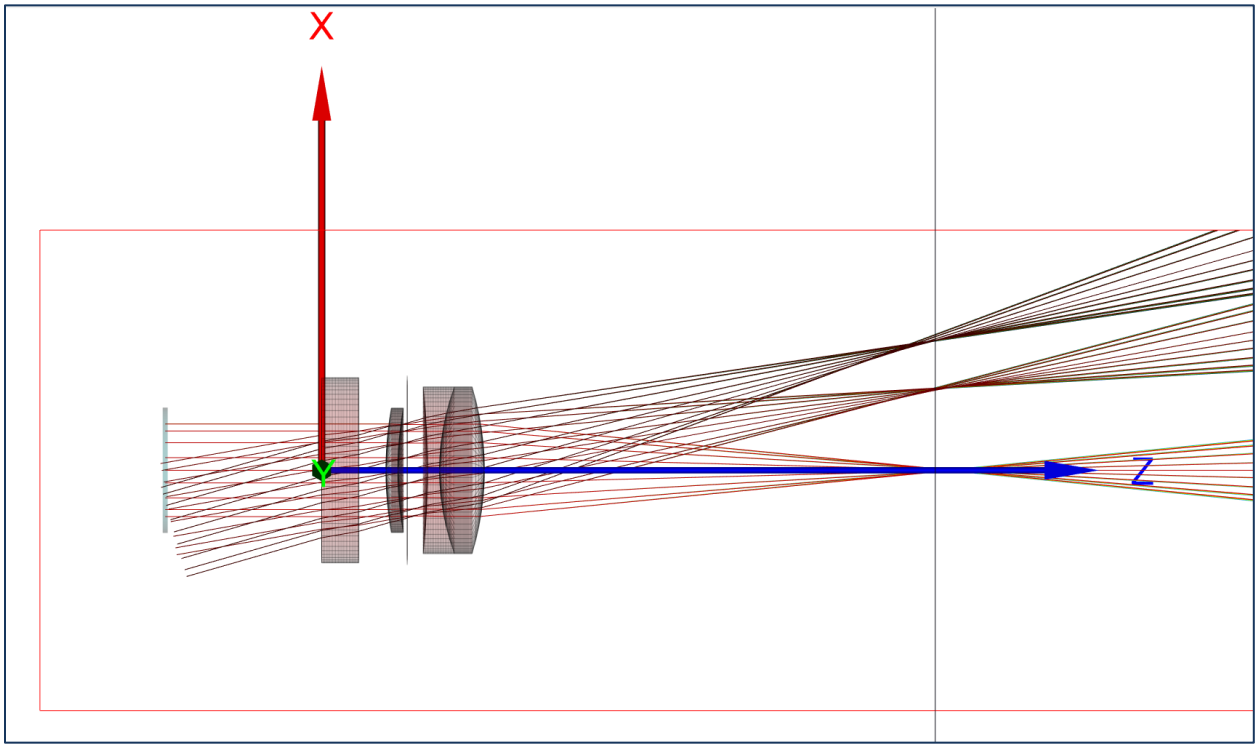
We're starting from the exact same initial conditions as the example above. Without considering the constraints of a Tessar-style lens which may change the sequence of our design, we know from experience that the doublet is a critical first step to get into a realistic range before designing the other elements. So, we'll first optimize surfaces 6, 7 and 8 which are the three surfaces of the cemented doublet, within the default bounds chosen based on our lenstack input:



Curiously, that resulted in a doublet very similar to a Tessar-style. That was likely due to our choice of materials for the two elements of the doublet and our placement of the stop very near the center of the lens.

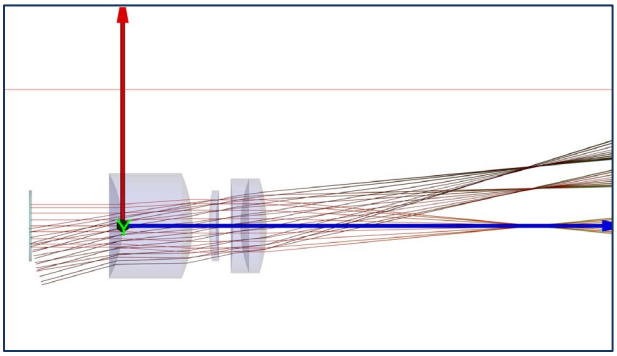
As far as convergence on our image plane, this first step driven by intuition went very far toward reaching plausible convergence even at the off-axis angles.

Our center lens is also a troublesome one at times so as a second step, let's optimize it's two surfaces, again, after only seconds, we yield the following:

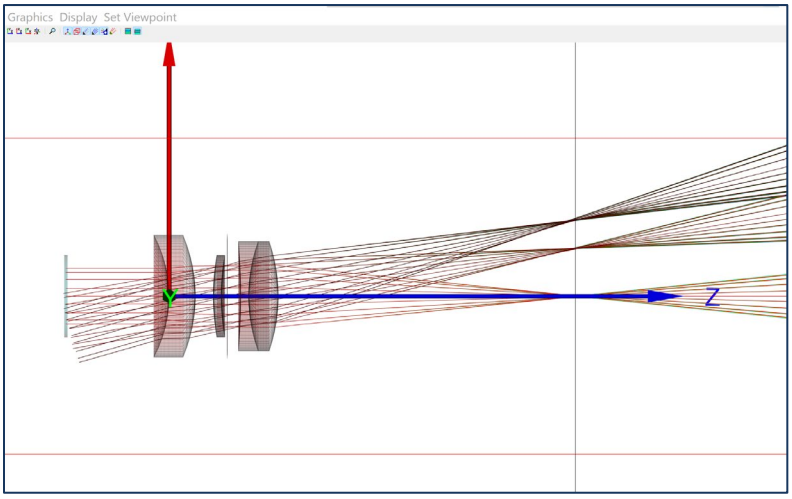


That did not improve nor detract from our design progress. Our experience tells us that we may want to try different materials with a greater difference in index of refraction between the two materials of the doublet, or possibly alter the curvature of the center lens to improve focus off-axis. We know we have aberration issues but aren't at a sufficiently advanced stage to deal with them yet. For no other reason than it being the final lens to rough out at this early stage of design, we decide to turn our attention to the front lens.

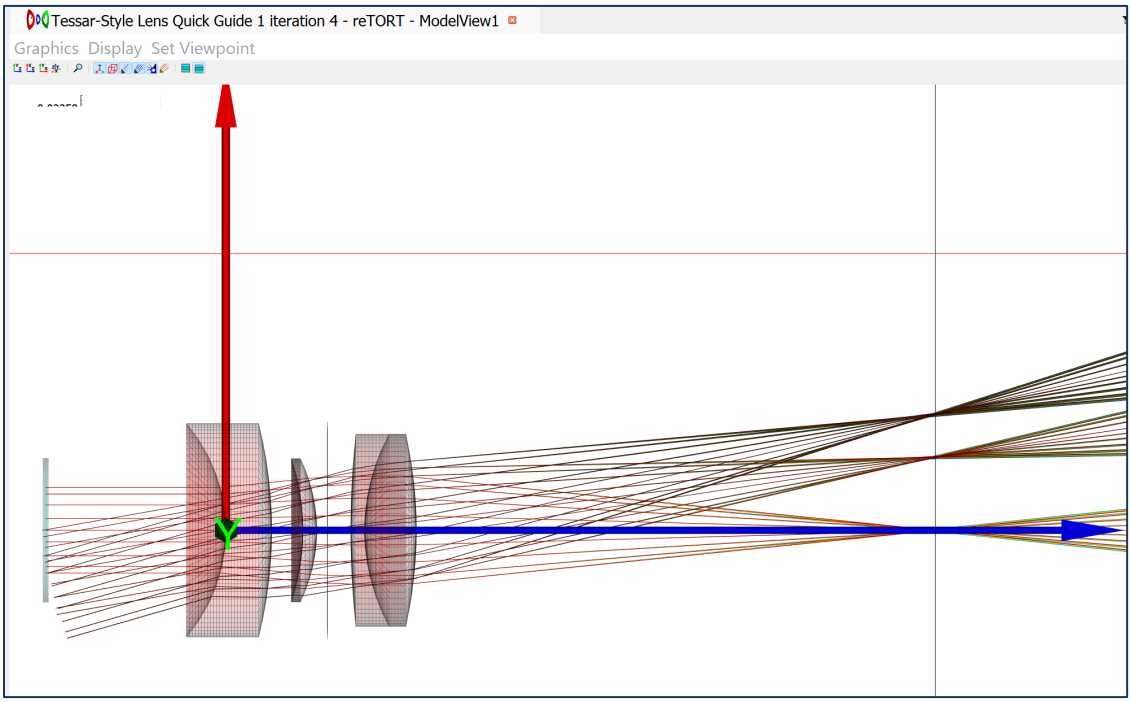
This time, in hope of making a larger step in the right direction, we decide to optimize both surface 1 and 2 as well as the thickness of the front lens. Again, our global optimizer provides a solution in less than a minute but it's a surprise:



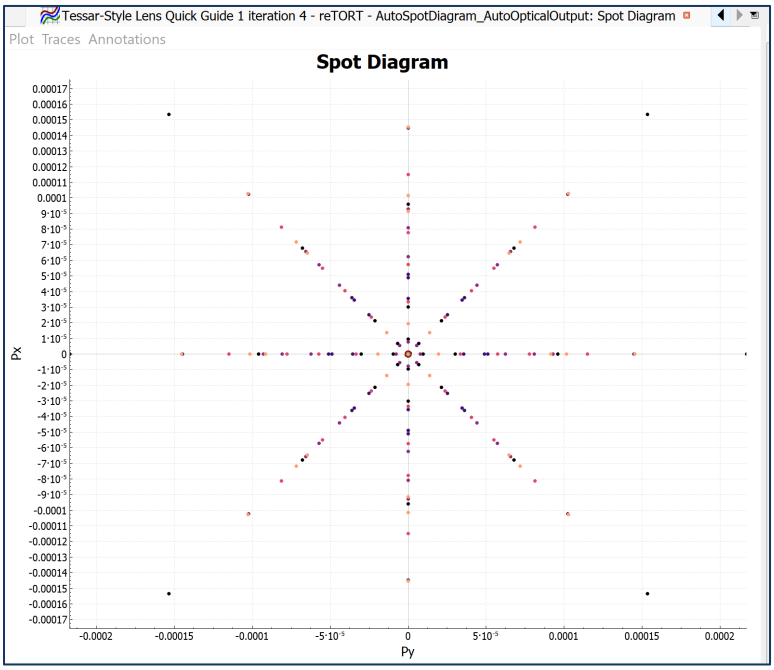
This is our step backwards, that element is way too heavy and a waste of glass. This step was obviously not the right direction although our 16 degree axis ray bundle did move slightly closer to convergence on the image plane. We do believe the curvatures are workable and so we decide to manually reset the thickness, retaining the optimized curvatures, and go back and re-optimize the doublet. We set the thickness of the front less to what we know to be a reasonable value of 0.3 inch and run the global optimizer again to yield:



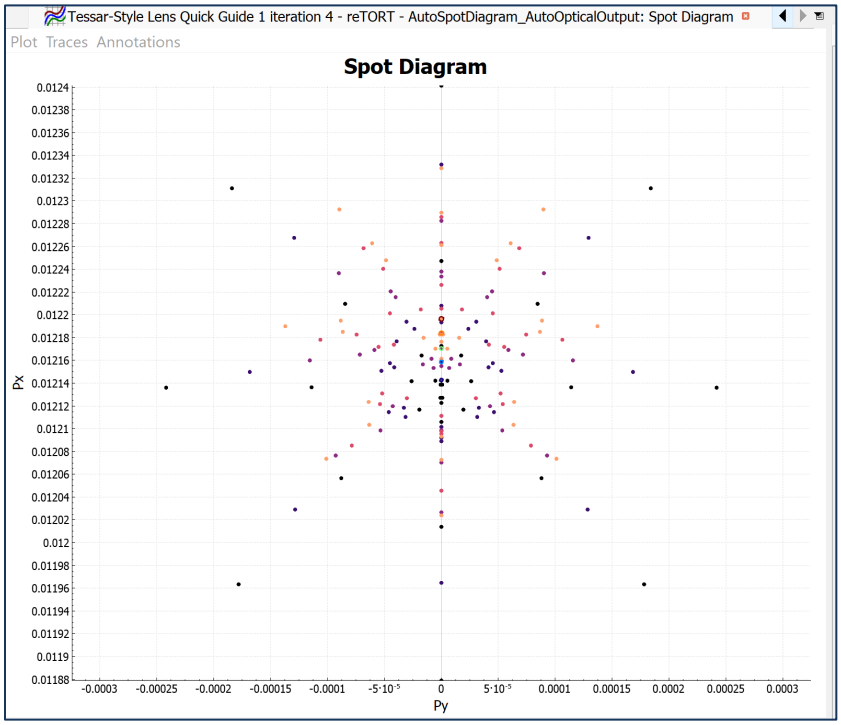
We're getting very close and our ray traces have been stable and resulting in good cost levels. We've advanced the initial conditions to a point where our bounds can be better defined and we believe we can optimize all surfaces at once without being hung up trying to exceed our bounds:



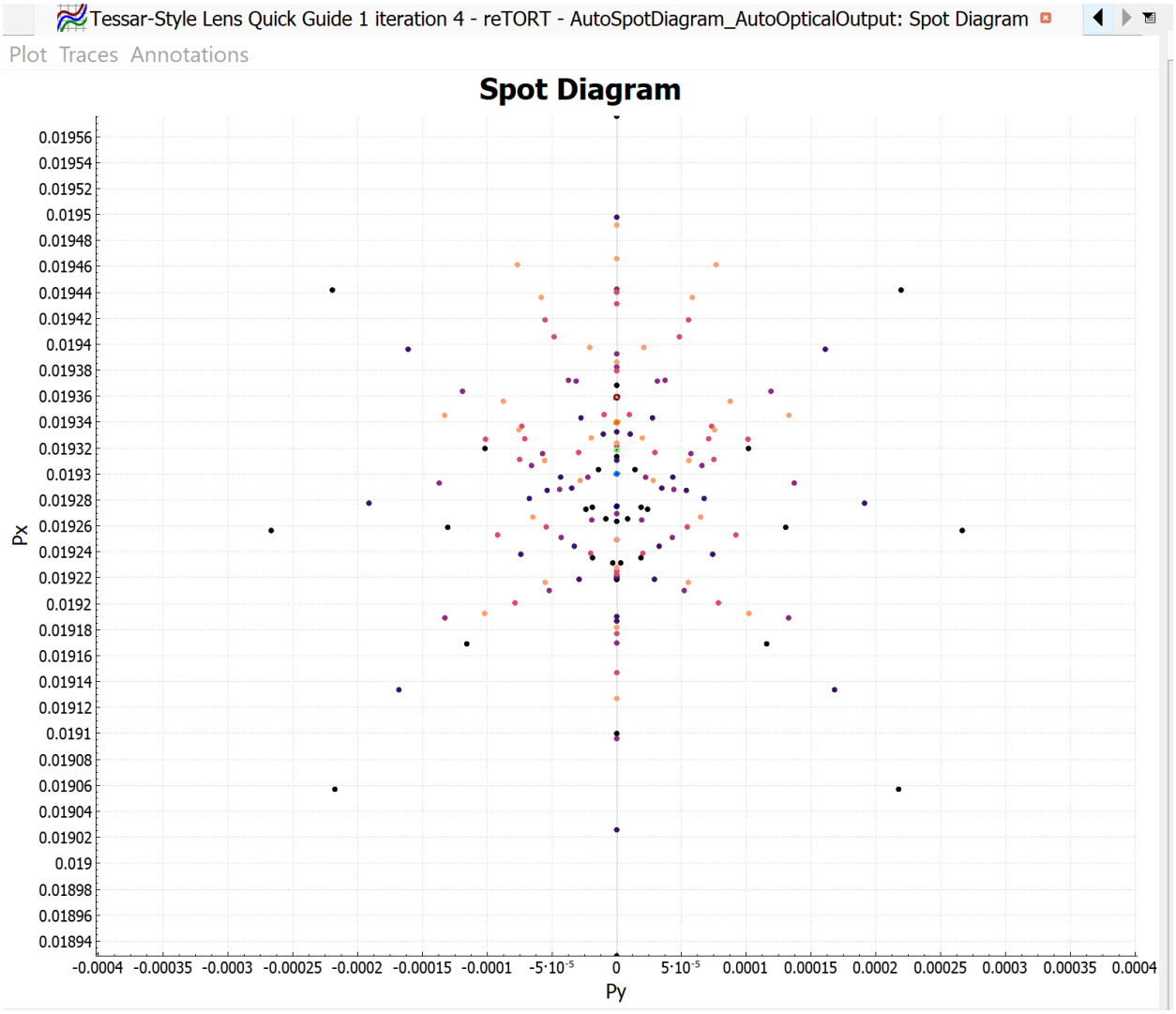
This looks like a major improvement, let's look at the spot diagrams, first on-axis:



Looking good, now the 10 degree off-axis which also looks very reasonable for this early stage of design:

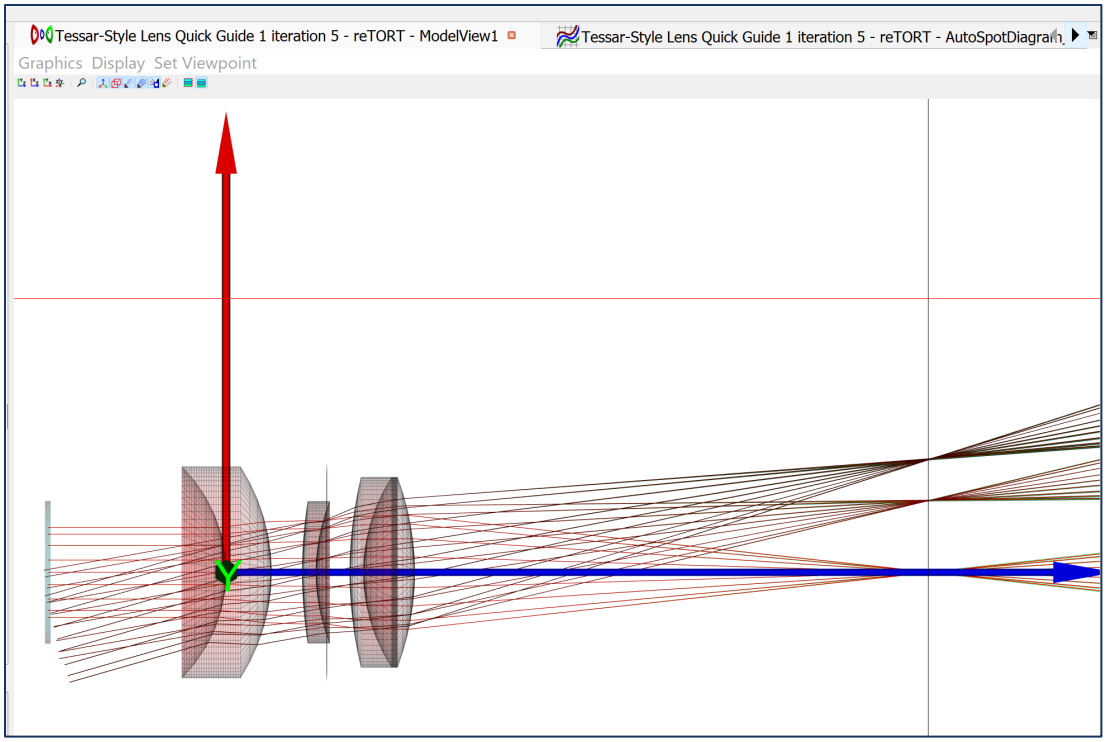


Finally the 16 degree off-axis:



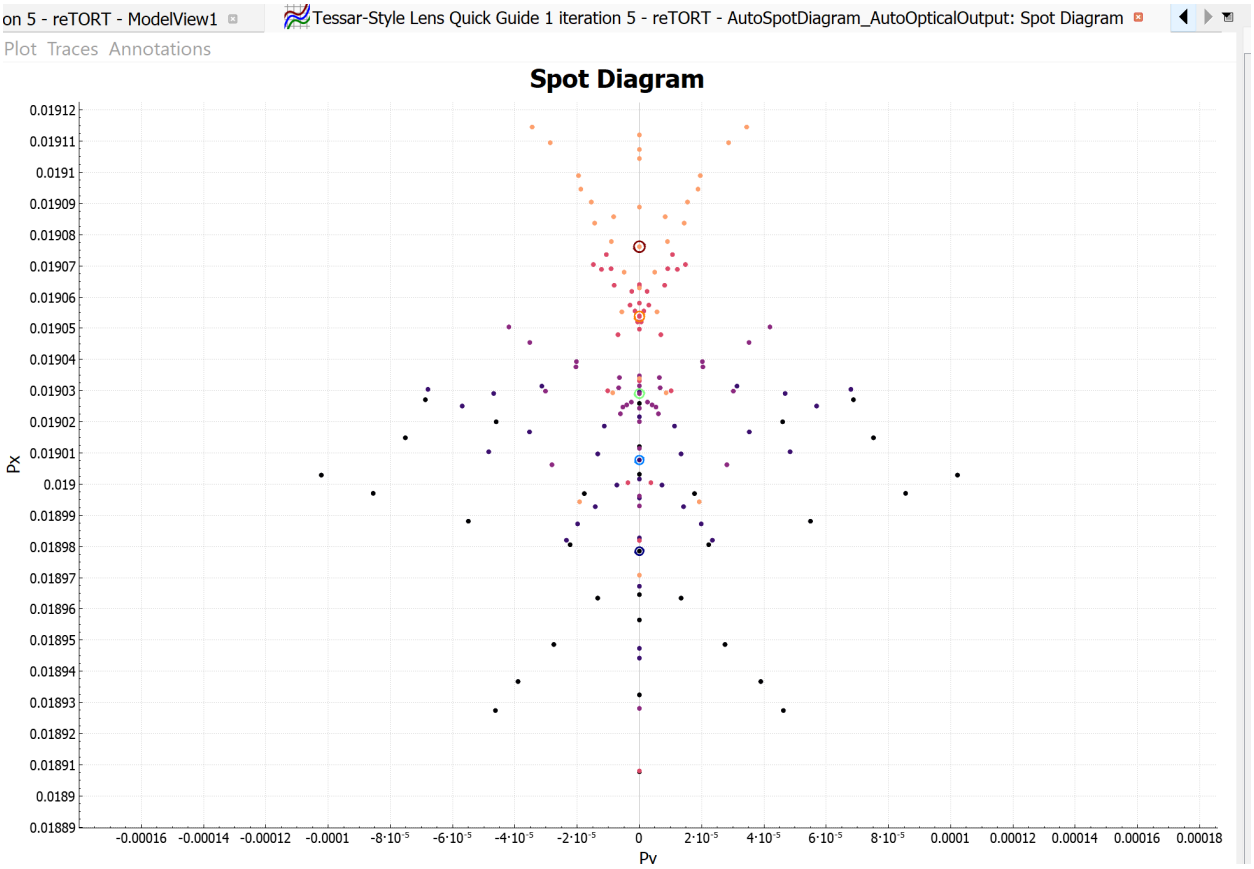
This too is very workable and we can proceed to further analysis and reduction of aberrations with this design, as well as tolerance analysis.

But despite the speed and accuracy of the E x H global optimizer, at this time, we sometimes try our DLS local optimizer to determine if there's a better performing and lower cost solution nearby. Often this step is determined by the closeness of our global optimized solution to its preset boundaries as DLS will be allowed to step slowly outside those boundaries. That really is not the case here but let's run DLS and compare:



This optimization took only 1.2 seconds.

And at first blush, it does appear we've improved convergence on the image plan. Let's take a look at the 16 degree off-axis spot diagram:



This is not an improvement and our final globally optimized solution is gong to be our candidate for further analysis. But let's save this GEMSIF file and the results as a backup in case further analysis indicates that we should consider this approach. This was just an addendum to our prime purpose of learning how to navigate reTORT/GEMSIF. A future design guide will get into that next step of analysis.

What we did learn in this addendum exercise though was that an iterative process can yield comparable optimization results using the E x H global optimizer while enforcing bounds in 10% to 20% of the time that a more comprehensive optimization approach, global or local, will take.

Useful tips in using reTORT/GEMSIF

These are taken from the last section of our feature list, available at https://exhsw.com/retort-ray-tracer/#featurelist.

1. Always pay attention to the status bar, It will warn you of problem situations and save you valuable time

2. Save interim satisfactory designs with descriptive names, You can always come back to that branch of your work and take it in a different direction

3. When you cannot find a function, follow the Model Hierarchy, it follows a typical analysis, starting with the Optical Source and ending with Results and Views

4. When you cannot find a property, click on the Model Hierarchy among the functions in which you're interested and then scan the Property Editor window that opens, you'll soon learn where to quickly find every bit of design data

5. Make use of the Parameters Dock to leverage your time by defining additional parameters that can be used by multiple functions

6. Until you become expert in reTORT and GEMSIF, when you have questions, always click on the function in which you're interested and look to the documentation dock in the lower right for further definition. Usually, you're curiosity will be satisfied by the first paragraph you see.

7. Delays in optimization are usually due to the solution space going outside of the optimization bounds. In this case, it is pulled back in and reattempted. In some cases, this can happen many times. We will provide messaging to the user in the near future that this has occurred so the user can decide whether to adjust their optimization bounds.

8. You may experience delays in stopping an optimization. The GUI including all rendering occupies one logical core, so long as more than one is made available to the program. All optimizations occupy the balance of the logical cores. When you issue a GUI command to stop an optimization, currently, the optimization is not interrupted frequently enough to receive that command from the GUI and kill all tasks in what many users would find a sufficiently short time. In the near future, we will improve this issue without affecting optimization speed. For now, if you have this issue, uncheck enforce global bounds unless you have critical constraints that must be enforced to realize your design objectives.