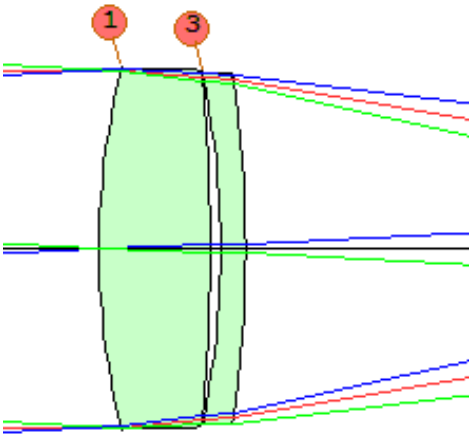
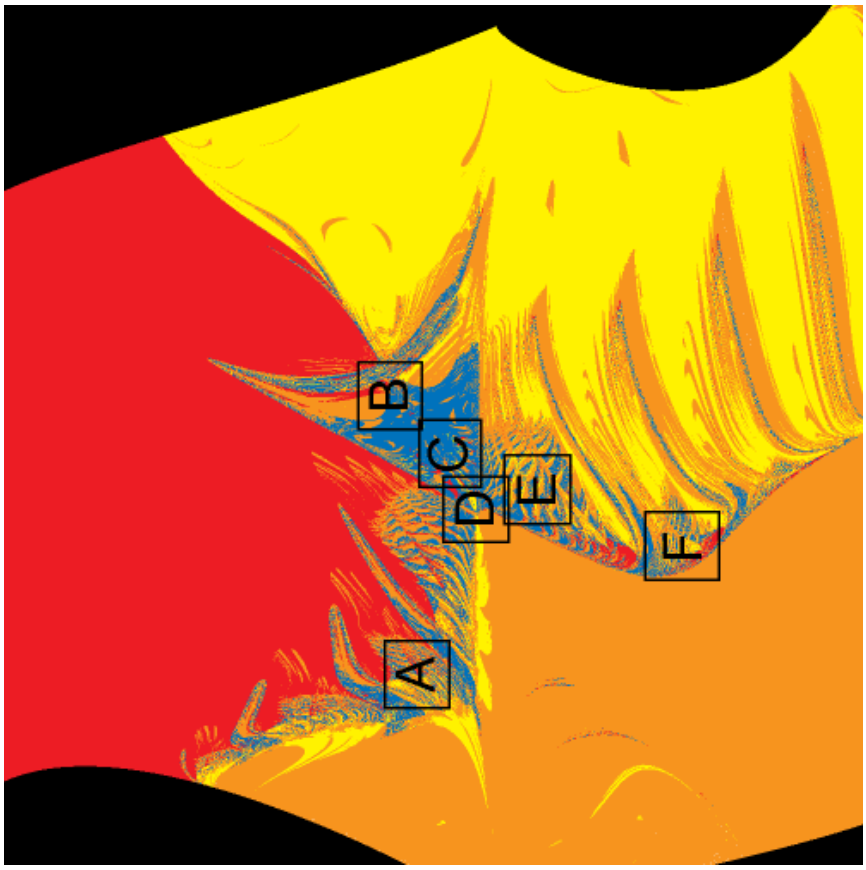
Lesson 23: Parametric Optimization Study + Ray Failure Correction

In this lesson we will explore a powerful but seldom-used feature of SYNOPSYS: it can do parametric studies showing the effects of two variables on a third. In this case, we want to see how the end point of a lens optimization run depends on the starting point. In a perfect world, every starting point would go to the best of all possible optima, but the world is not perfect yet. There are usually many local minima for any given problem, and the best we can expect is that a good optimization algorithm should go reliably to the closest one. (There are of course so-called *global* optimization algorithms, such as DSEARCH, but that is a different topic. Here we will analyze the process that simply minimizes the merit function, starting with a given configuration.)

One would therefore expect that two starting points that are almost exactly the same would go to the same local minimum, even if it is not global. How well do current algorithms perform on this score? Some very interesting results have been discovered by Dr. Florian Bociort of TU Delft. He ran a simple case, the doublet shown below.

To make the job very simple, he corrected rays at three field points in the major color only, ignoring edge violations. Then he varied the starting value of radii 2 and 3 in a raster fashion and made a plot where the color of each pixel on the grid encodes the final value of the merit function. He found that there are several local minima, which is not surprising, even for so simple a problem—but what was completely unexpected was how the merit function varied in a chaotic manner in many places. Thus, nearby starting points often go to very different end points. (He did this analysis on Code-V.) Here is a figure from his article at [http://homepage.tudelft.nl/q1d90/FBweb/fractals.html.](http://homepage.tudelft.nl/q1d90/FBweb/fractals.html)





(We have turned this picture on its side so it will line up with the SYNOPSYS analysis, below.)

Notice how the results near the boundaries of the zones of attraction are complex and chaotic. The black areas show starting points that gave ray failures and therefore could not be analyzed.

We suspected that the PSD algorithm in SYNOPSYS was more reliable and stable than the method used for the above picture, so we set up a run on the 3-parameter evaluation feature PA3. Here is the input:

Starting doublet:

```
RLE
ID FLORIAN STARTING DOUBLET
WA1 .5876000
 WT1 1.00000
 APS               1
 UNITS MM
 OBB  0.000000    3.00000   16.66670    0.00000    0.00000    0.00000   16.66670
   0 AIR
   1 CV      0.0146498673770   TH     10.34600000
   1 N1 1.61800000
   1 GID 'GLASS           '
   2 RAD   -174.6512432672814   TH      1.00000000 AIR
   2 AIR
   3 RAD    -80.2251653581521   TH      2.35100000
   3 N1 1.71700000
   3 GID 'GLASS           '
   4 RAD   -111.8857786363961   TH     92.41206276 AIR
   4 AIR
   4 CV      -0.00893769
   4 UMC     -0.16667000
   4 TH      92.41206276
   4 YMT      0.00000000
   5 CV      0.0000000000000   TH      0.00000000 AIR
   5 AIR
 END
STORE 5
```

And this is the input for the PA3 program):

```
ON 78              ! use finer grid (118x118 points)
PA3 LOOP COLOR          ! initialize PA3, request color boxes for output
RZ1 -.025 .04           ! set the range of variable Z1
RZ2 -.045 .075          ! set the range of Z2
RZ3 0 5.5          ! display results over this range of merit function values
NOSMOOTH                ! there will be steps in the output; do not smooth
XLAB "2 CV -.025 .04"   ! define the label for the X-axis, which is variable Z1
YLAB "3 CV -.045 .075"  ! label for Y-axis, Z2
ZLAB "MERIT"            ! label for Z-axis, the final merit function
LOOP               ! tell PA3 to loop over the above raster of data

GET 5              ! get the starting lens each time
2 CV = Z1          ! set curvature 2 to the value of variable Z1
3 CV = Z2          ! and CV 3 to Z2, using the artificial-intelligence parser

PANT               ! initialize the variable list
VLIST RAD 2 3      ! and vary two radii
END                ! end of the variable list

AANT               ! initialize the merit function definition
```

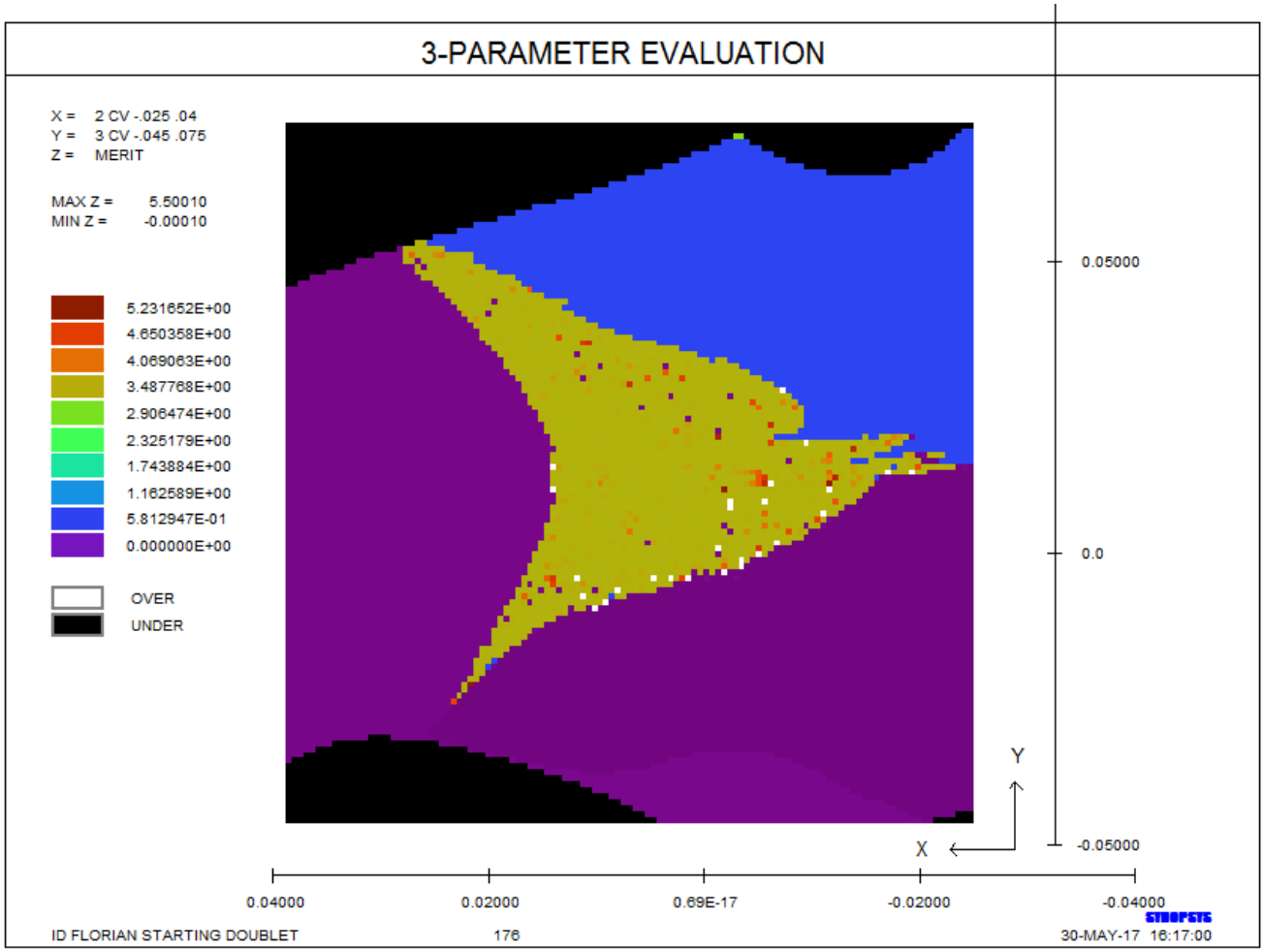
```
GSR .5 10 3 P 0          ! correct a sagittal fan, three rays, on axis
GNR .5 1 3 P .75         ! correct a full grid of rays, primary color, 0.75 field point
GNR .5 1 3 P 1           ! same, at full field.
END                 ! end of merit function definition

DAMP 10000          ! initial damping (see below)
SNAP 50                  ! watch what happens, but not too often, in order to keep it fast
SYNOPSYS 100             ! optimize until it converges

Z3 = MERIT          ! assign the current merit function value to variable Z3
PA3                 ! tell PA3 to cycle to the next case.
```

Why the high damping? (The default is 1.0 or 0.01, depending on mode switches.) The first iteration in SYNOPSYS is a DLS (Damped-Least-Squares) cycle, and we want to avoid any chaos that results on the first pass from that algorithm; the high damping will ensure that the lens changes very little on that pass. The more powerful PSD algorithm keeps track of changes in the first derivatives from pass to pass and deduces information about higher-order derivatives. That is the magic behind the PSD method, but it only works starting at the second pass.

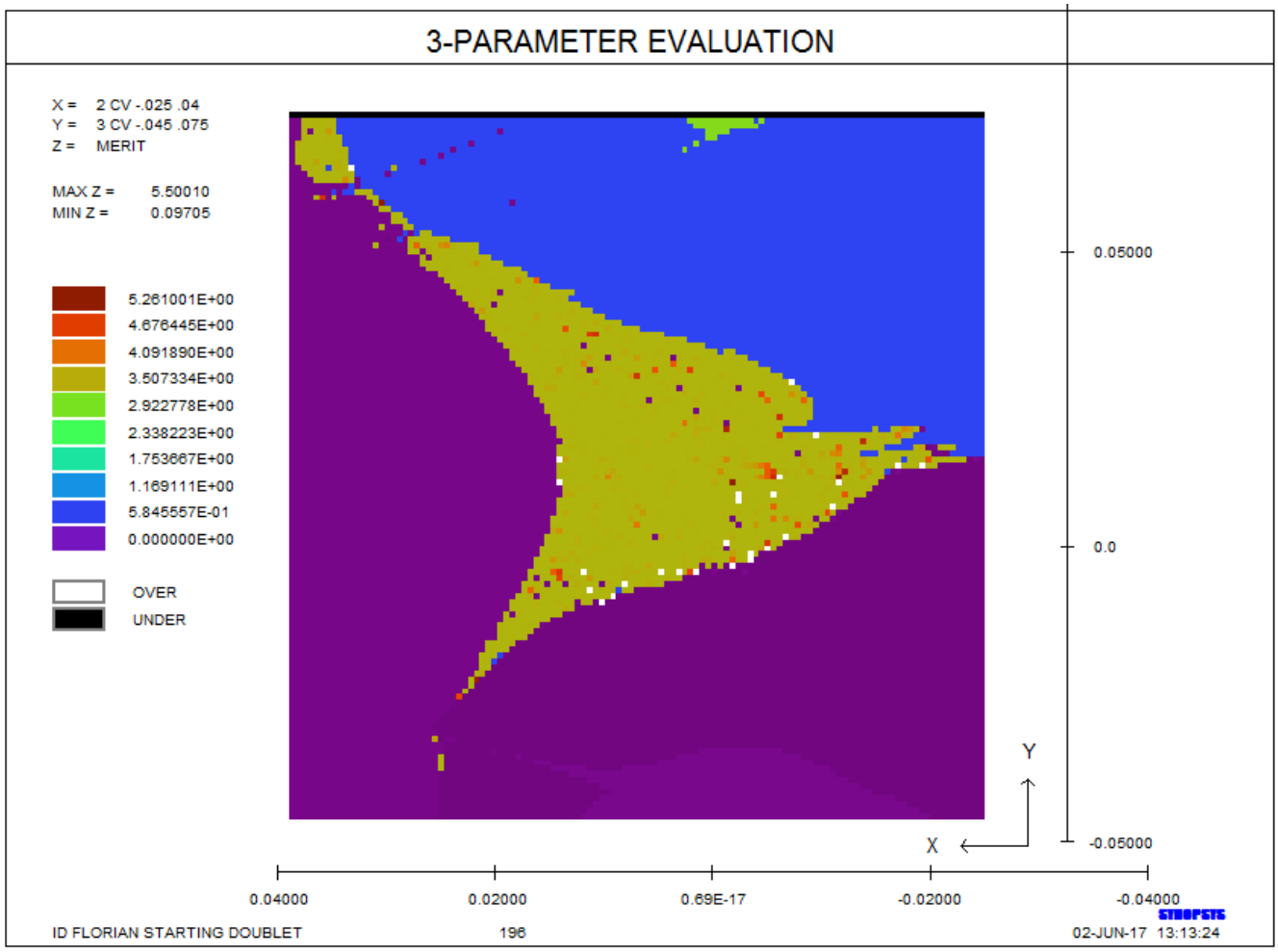
The results of this study are shown below. The violet areas on the left and near the bottom show that the program went to the same minimum for very different starting points—whereas in Florian's study those areas went to different minima. There is no evident chaos at the boundaries of the zones of attraction, as we expected would be the case with the PSD method, although scattered poles do show up in the central green region. We attribute the latter to the nonzero changes made by the DLS method on the first pass. Indeed, if we run this again with different initial damplng, those random spots appear in different places.



The black areas at top and bottom show where the starting points yield ray failures, in the same place where they did in Florian's study. We are curious what would happen if we activate the automatic ray-failure correction feature found only in SYNOPSYS™. We change the SYNOPSYS command to

`SYNOPSYS 100 0 FIX`

and rerun this job.



Now we see that the program has corrected the failures over every point where they occurred before. The starting lenses that could not be optimized by Florian now all yield respectable solutions. There is now some very slight chaos evident at the boundaries in areas that were previously all black, however, and we attribute this to changes the ray-failure correction program made to that starting point. Those changes sometimes moved the lens closer to another zone of attraction.

This very simple study involved optimizing with only two variables. What happens if we add CV 1 to the variable list? Try it and see! (The boundaries shift about somewhat, and the scattered spots no longer show up.)