**PROJECT**

***TEST OPTIMIZATION***

***STAT 6601***

***Summer 2017***

**Authors**

***Trung Phan***

***Devansh Pandit Vaishnavi Ayyadurai***

***Abstract***

*This project is about testing default OPTIM and NLM function in R for five functions namely, Beale, Booth, Matya, Rosenbrock & Easom. As part of testing, we are generating random numbers from -19 to 21 & all the functions are run 40,000 times on each function. OPTIM is run using default method Nelder-Mead on booth and Rosenbrock function using control list of absolute tolerance of 1e-16. Also, other methods like BFGS and CG are tested on other functions like Easom, Matya and Beale with control list optimization.*

*All the functions are being tested but the report consist of plots for functions highlighted in color amber which portrays good convergence.*

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Method | Functions | Proportion of times convergence was reached (%) | Times claim to  get converged | Times actually  get converged | | | Number of global minimum | Number of local minimum |
| err.norm  <1e-  5(OK) | err.norm  <1e-  7(Good) | err.norm  <1e-  9(Excel) |
| Nelder- Mead (Default) | f.beale | 97.545 | 39,018 | 3 | 0 | 0 | 1 | 0 |
| f.booth | 100 | 40,000 | 7 | 0 | 0 | 1 | 0 |
| f.matya | 100 | 40,000 | 2 | 0 | 0 | 1 | 0 |
| f.r | 99.98 | 39,992 | 1 | 0 | 0 | 1 | 0 |
| f.e | 100 | 40,000 | 3 | 0 | 0 | 1 | Few |
| BFGS | f.beale | 51.86 | 20,744 | 1,292 | 4 | 0 | 1 | 0 |
| f.booth | 100 | 40,000 | 40,000 | 40,000 | 16,738 | 1 | 0 |
| f.matya | 100 | 40,000 | 40,000 | 36,116 | 17,493 | 1 | 0 |
| f.r | 73.3575 | 29,343 | 466 | 8 | 0 | 1 | 0 |
| f.e | 99.9325 | 39,973 | 340 | 340 | 284 | 1 | Few |
| CG | f.beale | 4.5125 | 1,805 | 3 | 0 | 0 | 1 | 0 |
| f.booth | 100 | 40,000 | 40,000 | 0 | 0 | 1 | 0 |
| f.matya | 0.0975 | 39 | 0 | 0 | 0 | 1 | 0 |
| f.r | 0.0225 | 9 | 0 | 0 | 0 | 1 | 0 |
| f.e | 97.98 | 39,192 | 339 | 0 | 0 | 1 | Few |

***NLM:***

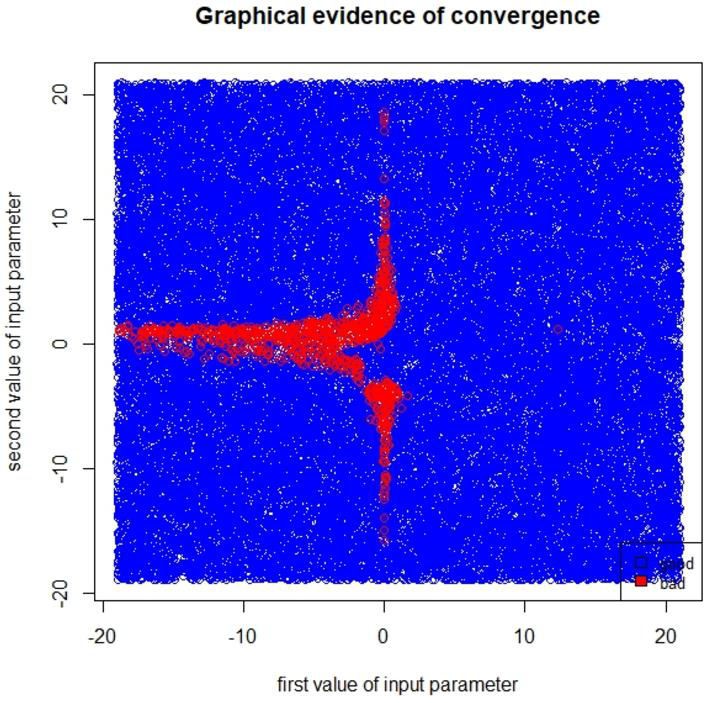
*All the functions are being tested but the report consist of plots for functions highlighted in color amber which portrays good convergence.*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Functions | Proportion of times convergence was reached (%) | Times claim to  get converged | Times actually  get converged | | | Number of global minimum | Number of local minimum |
| err.norm  <1e-5 | err.norm  <1e-7 | err.norm  <1e-9 |
| f.beale | 84.085 | 33,634 | 3,200 | 193 | 0 | 1 | 0 |
| f.booth | 100 | 40,000 | 40,000 | 10,687 | 3,912 | 1 | 0 |
| f.matya | 100 | 40,000 | 35,036 | 4 | 0 | 1 | 0 |
| f.r | 85.7575 | 34,303 | 27,265 | 77 | 0 | 1 | 0 |
| f.e | 12.4775 | 4,991 | 45 | 11 | 6 | 1 | Few |

**Nelder-Mead Method:**

**1. Beale Function:**

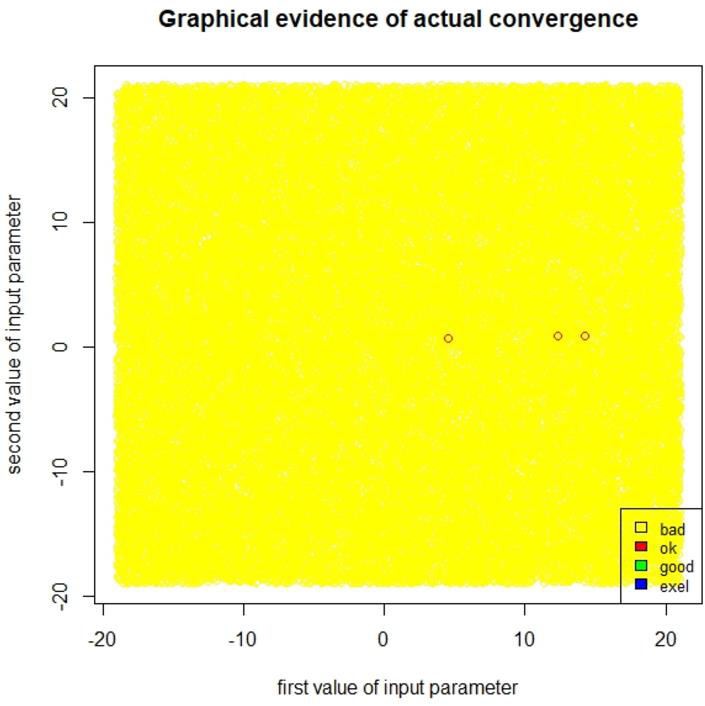
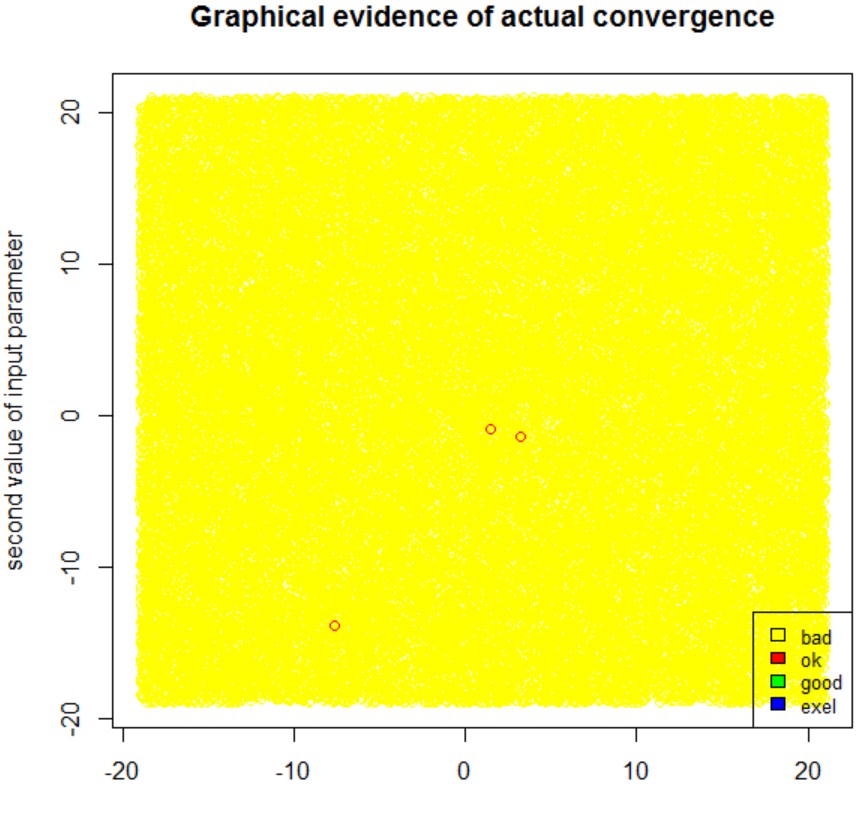
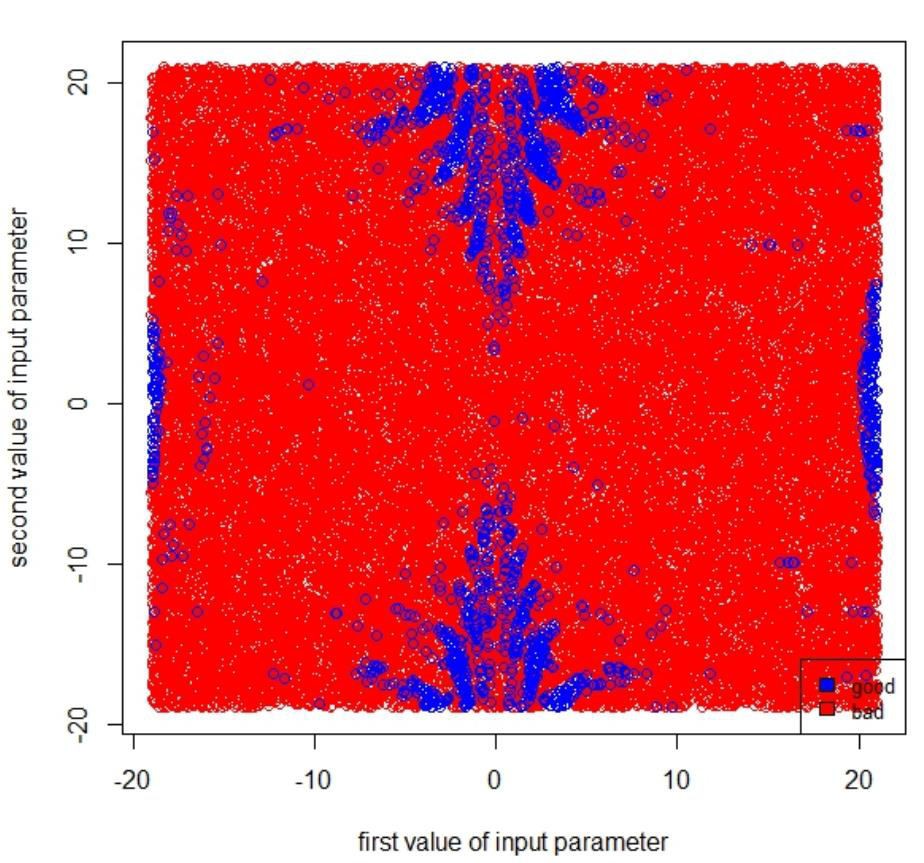
***A.******Evidence of Convergence:***



***Method CG for comparison:***

*Graph returns red points which do not get converged, while the blue points get fully converged. Most of un-converged pairs of input parameters tend to lie on the either x-axis and y-axis and their first value of input parameter almost got negative value.*

*Graph returns a lot of red points which do not get converged, while the blue ones gets fully converged. The number of converged pairs of input parameters become very less when reaching towards Original (0,0).*



***Nelder-Mead CG***

*There are 3 ok points in red color that got actually converged with err.norm<1e-5 for both default and CG methods.*

**B. *Initial & Final Values:***

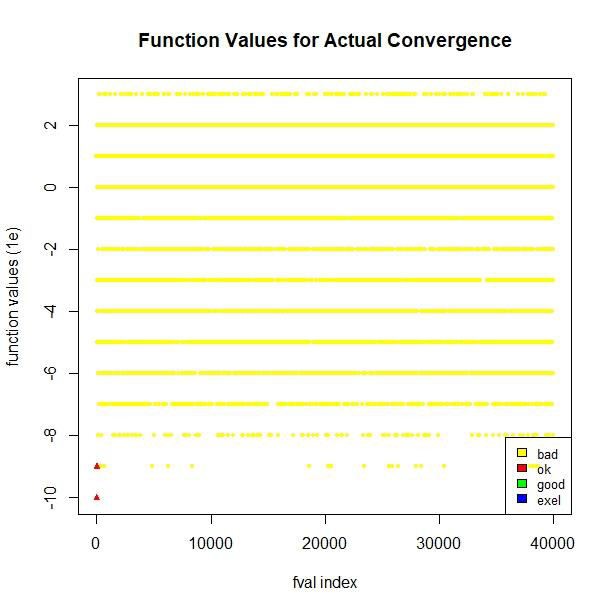


The function returned few optimal points. The points near to (3,0.5) where 3 is on x-axis and 0.5 is on y-axis are the optimal points. Rest of the adjoining point can be considered as slightly converged points. The function also returns several points which

are not converged & these points are near (-600,0) and (0,-250).

Page | 5

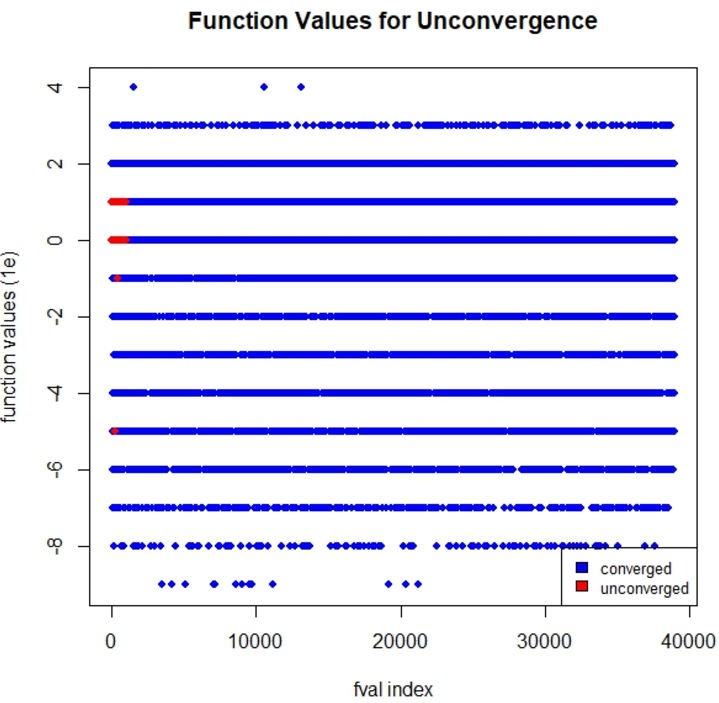
**C. Function value for actual convergence:**



***D. Function Values for un-convergence:***

*We have three ok points in color red which are very close by and have overlapped in the graph. They have very small values which are less than*

*1e-8*

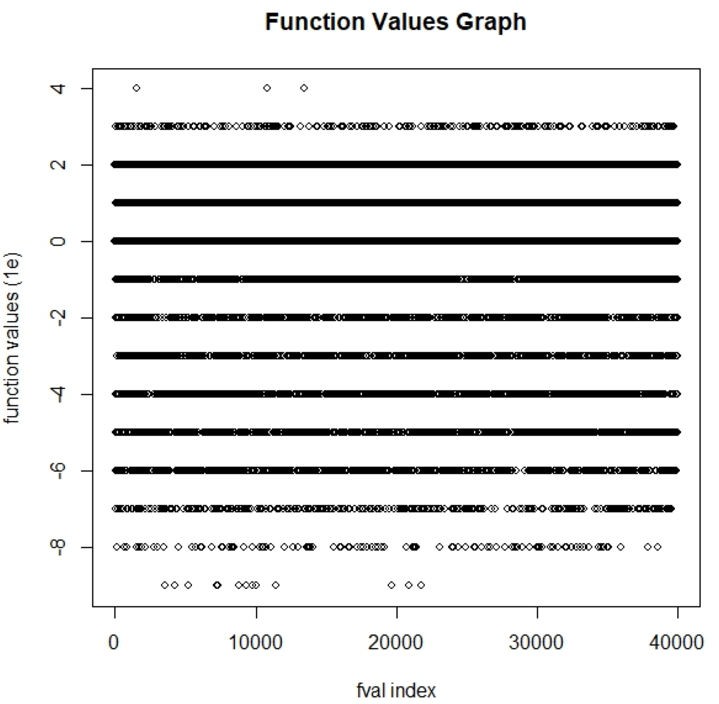


*We can see that there are some red points which the function claims that those are un-converged points. At those points, optim() will return the*

*non-zero convergence values.*

Page | 6

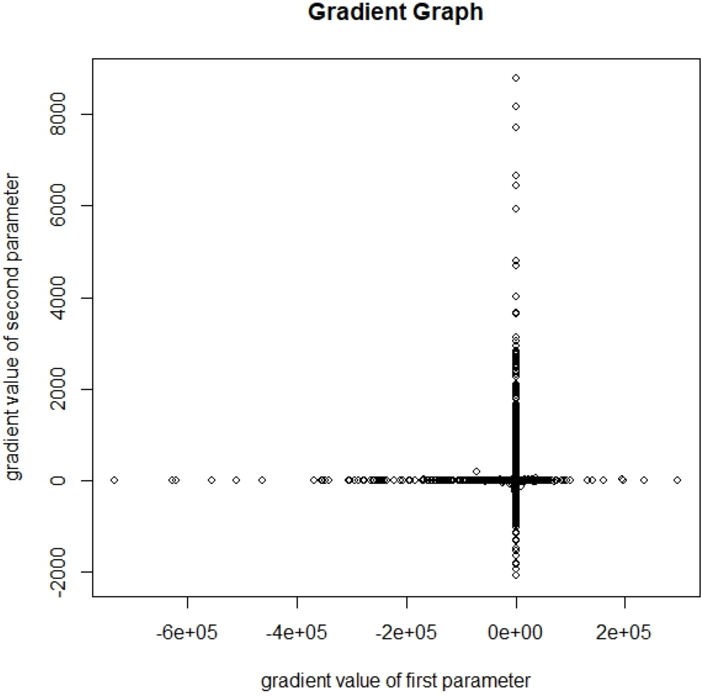
***E. Function*** ***Values:***



*There are few points lie on the line y= -8 and very few below that, which are the best values, there are other points(above -5 ) that are not good and seem to*

*be dispersed apart.*

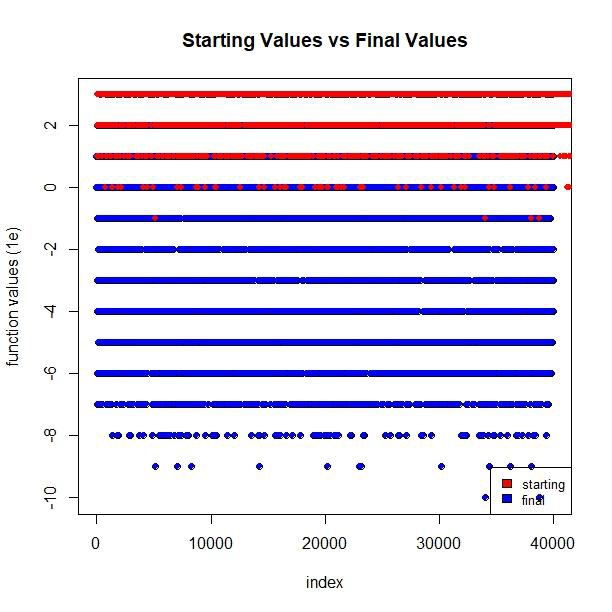
***F. Gradient Graph:***



*For optimal solution, the gradient should be zero. From the graph, lot of values seem to converge to zero and few points far away from zero, with values like 8000* *and low points with -2000.*

Page | 7

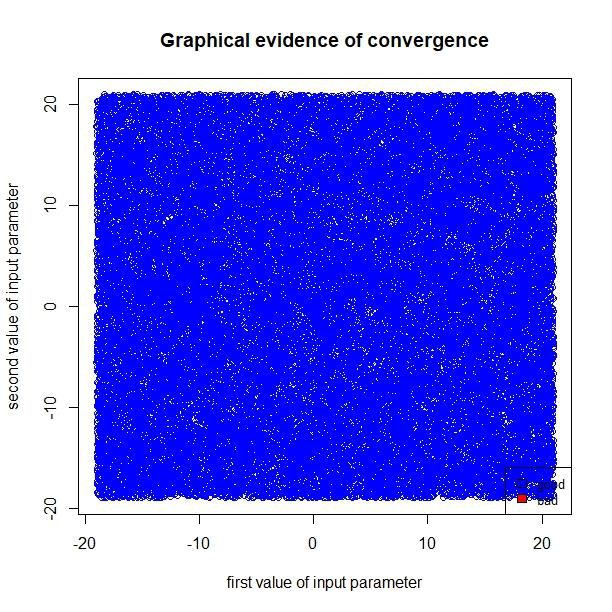
***G. Starting Values Vs Final Values:***



**2. Booth Function (Nelder-Mead):**

*The final values return a wide range of function value. There are some good final values attain function values that are very small (1e-10), nearly to 0.*

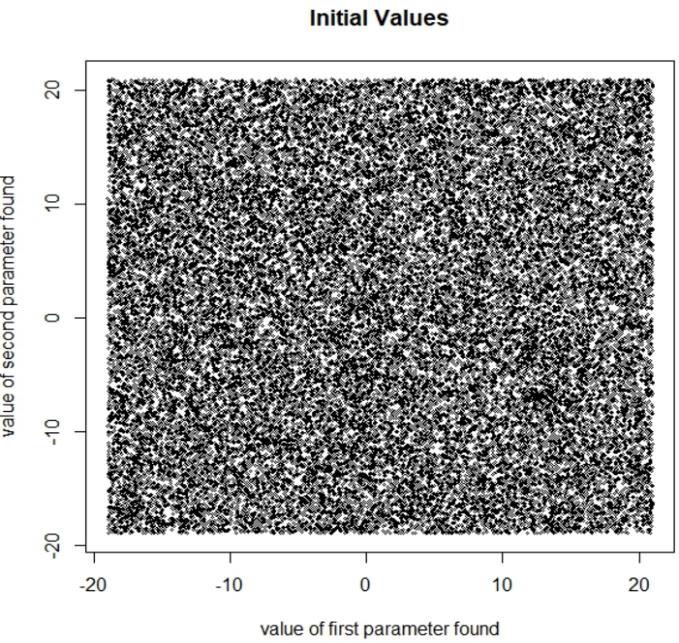
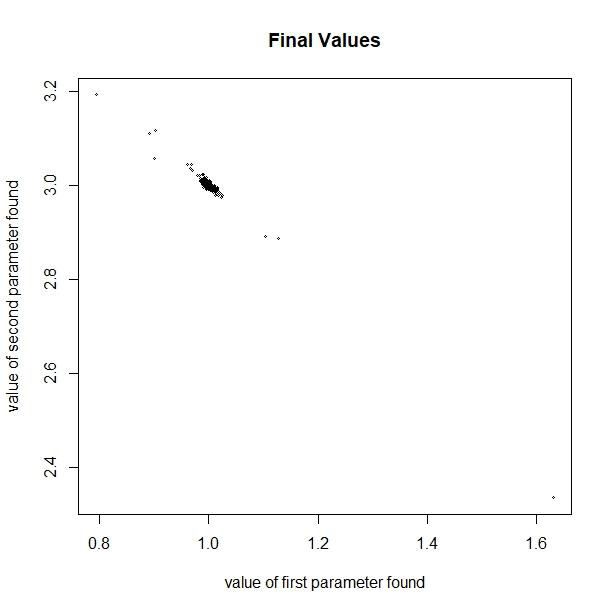
***A. Evidence of Convergence:***



*Optim() claims that 100% of the result are totally converged.*

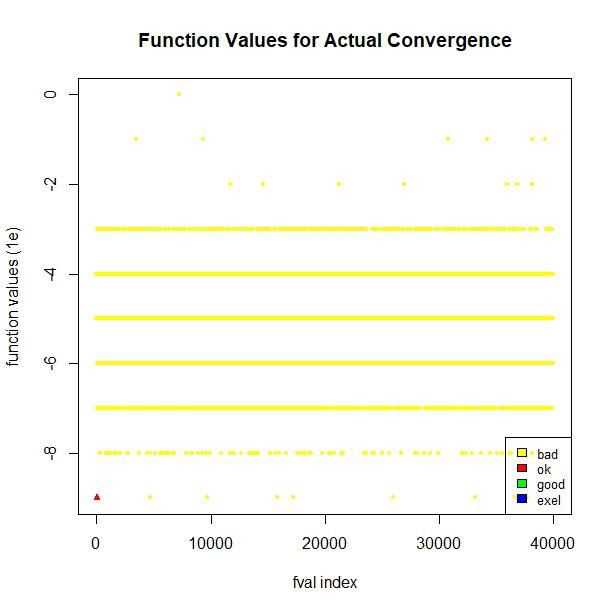
Page | 8

**B. *Initial & Final Values:***



*The function returned the output parameters gather near the real optimal point (1,3). Surprisingly, these points are not actually get converged. That means that optim() claims to get optimal minimum each time it run, unfortunately, these point are not significant.*

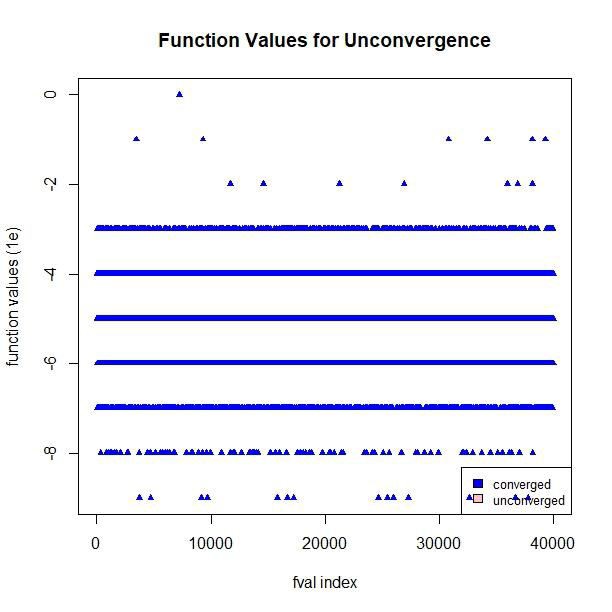
**C. Function value for actual convergence:**



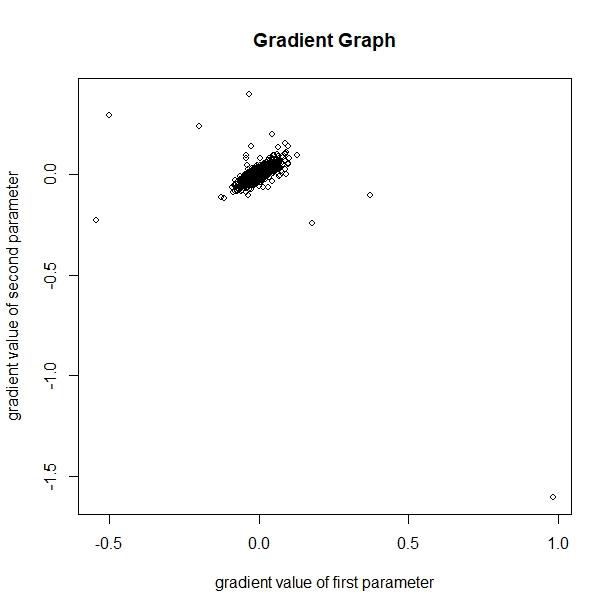
*There is only one point that get actually converged. This proves that optim() claimed to get converged 100% are not* *reliable. The FV (function value) of converged point is quite small around 1e-9*

Page | 9

***D. Function Values for un-convergence:***



***F. Gradient Graph:***

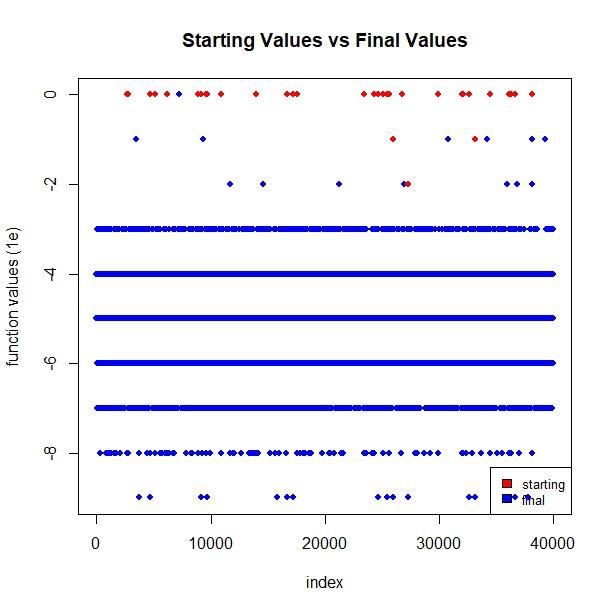


*By using default method, optim() claimed that all of the input parameters are totally converged. Therefore, there is no un- converged point in graph. We can also see that most of points that nlm() claim* *to get converged achieved FVs in range from 0 to 1e-8. Although they actually do not converged, their FVs attain at very small number. There are some point whose FVs even go below 1e-8.*

*For optimal solution, the gradient should be zero. Most of the time, optim() returns number of pair of parameters that get their gradient values near to (0,0). However, we can see that there are a few point are away from 0. The furthest point is near to (1, -1.5).*

Page | 10

***G. Starting Values Vs Final Values:***

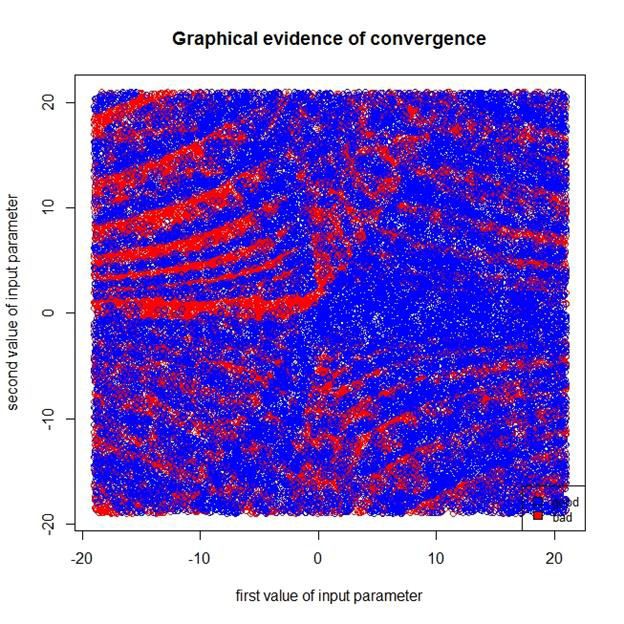


**BFGS Method:**

*The final values return a wide range of function value. There are some good final values with function values that are very small (below 1e-8), which are nearly to 0.*

**1. Beale Function:**

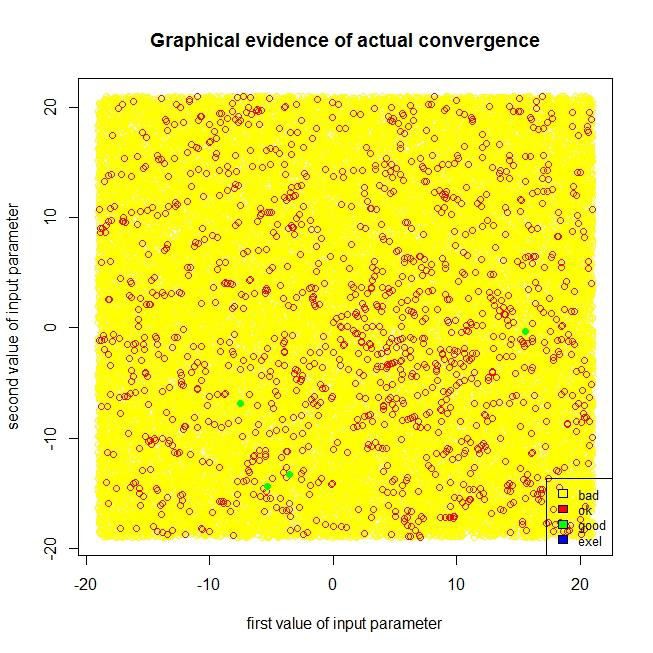
***A. Evidence of claimed Convergence***



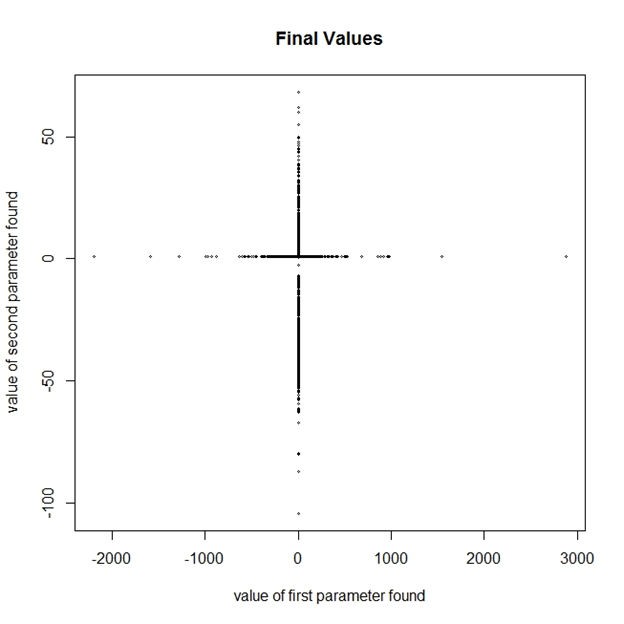
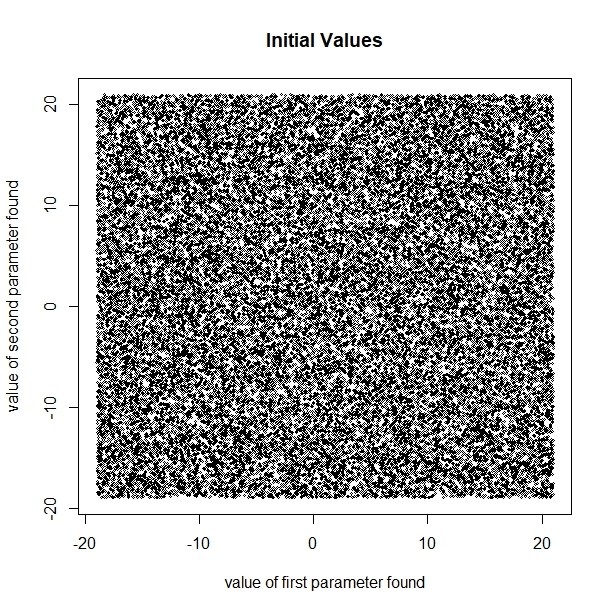
*In the claimed convergence, we have lot more (blue points) good points. All the un- converged pairs of input parameters are red points.*

Page | 11

***B. Evidence of actual Convergence:***



***C. Initial & Final Values:***



*Graph returns yellow points which doesn’t get converged, while the red points(ok) get converged. We also got some good points (in green ).*

*The function returned few optimal points. The points near to (3,0.5) where 3 is on x axis and 0.5 is on y axis are the optimal points. Rest of the adjoining point can be considered as slightly converged points. The function also returns several points which are not converged & these points are near (-600,0) and (0, -50).*

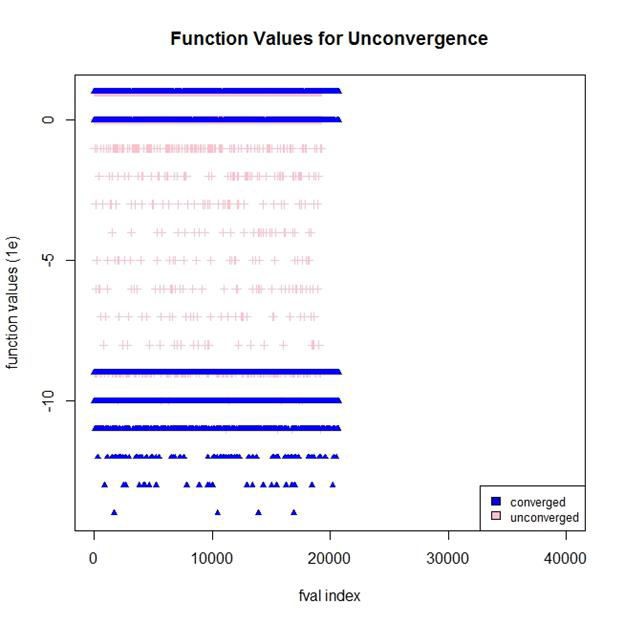
Page | 12

**C. Function value for actual convergence:**



*We have some “ok” points in color red near or below 1e-10 which are very close by and have overlapped in the graph. We also have 4 ok points which are marked as green triangles. Surprisingly, all of the actual converged points can be found at a very early stage.*

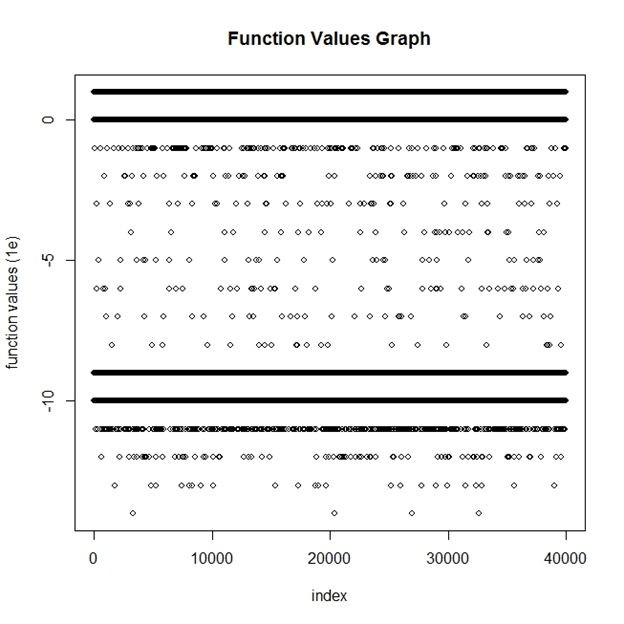
***D. Function Values for un-convergence:***



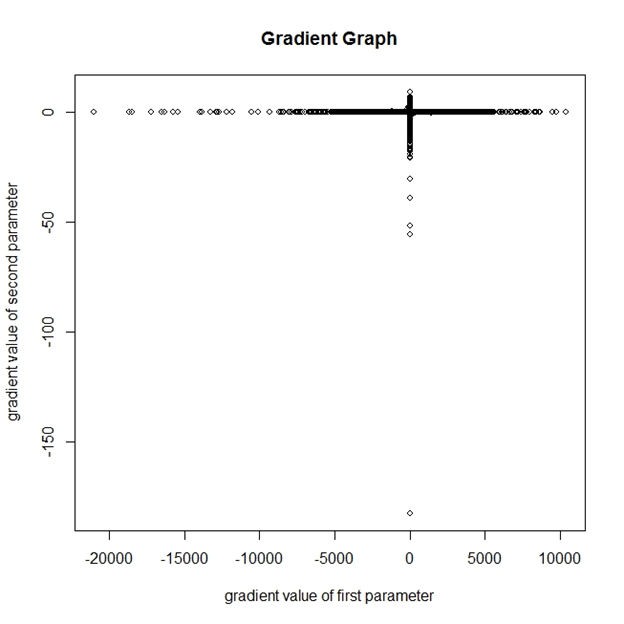
*We can see that there are some pink points which the function claims that are un-converged points. At those points, optim() will return the non-zero convergence values.*

Page | 13

***E. Function Values:***



***F. Gradient Graph:***

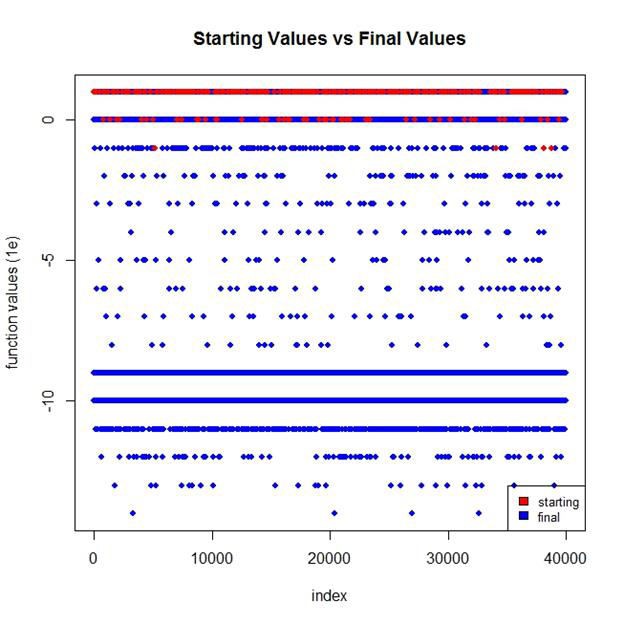


*There are multiple points lie on the line y=-9 and various below that, which are the best values, there are other points (above 0) that are not good.*

*For optimal solution, the gradient should be zero. From the graph, lot of values seem to converge to zero and few points far away from zero. Clearly, the gradient values for first parameter vary in wider range than that of the second parameter.*

Page | 14

***G. Starting Values Vs Final Values:***

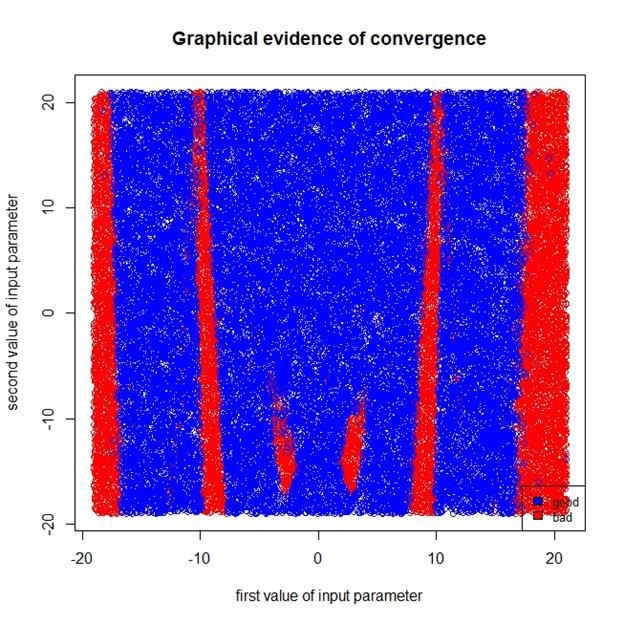


**2. Rosenbrock Function (BFGS):**

*The final function points converge mostly near 1e-9 and 1e-10. There are some good final values plotted in blue dots with function values that are very small (below*

*1e-11), which are nearly to 0.*

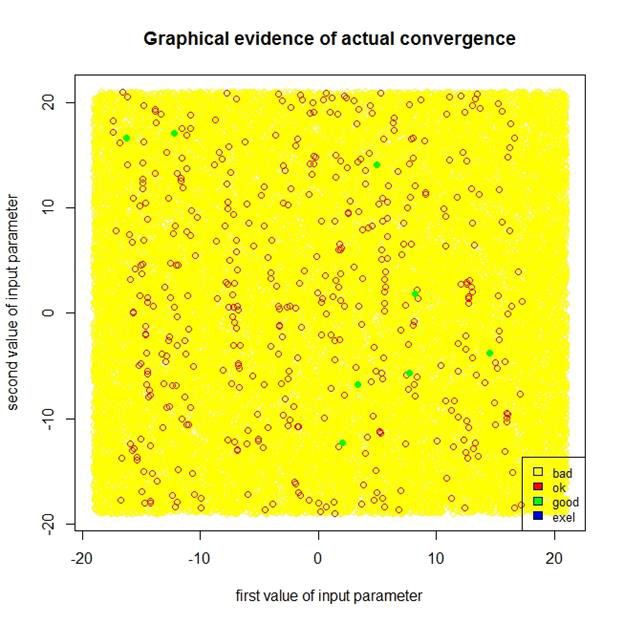
***A. Evidence of claimed Convergence***



*In the claimed convergence, we have lot more converged points (blue) than un-converged points (red) . Most of un-converged points are following in same pattern which parallels with the y-axis.*

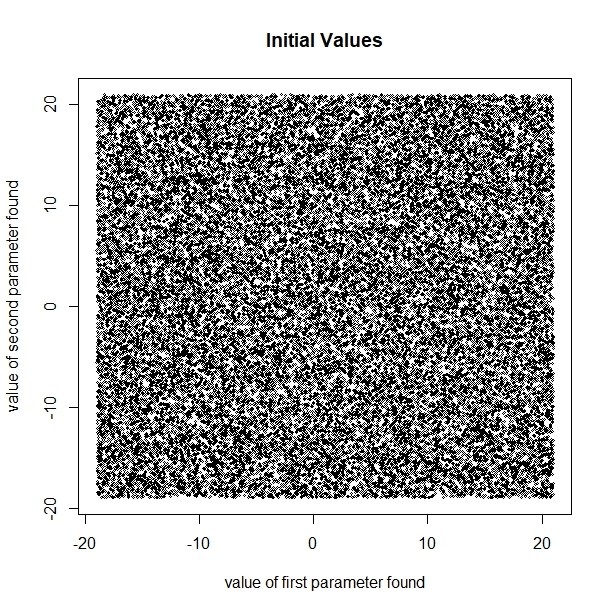
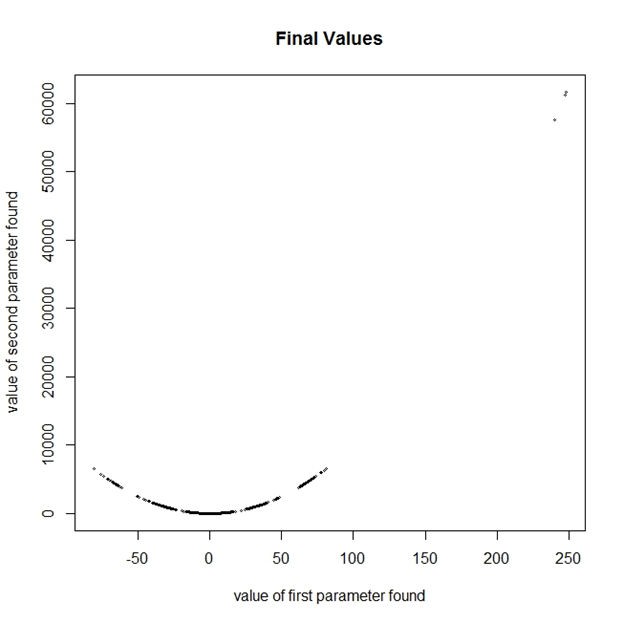
Page | 15

***B. Evidence of actual Convergence:***



**C. *Initial & Final Values:***

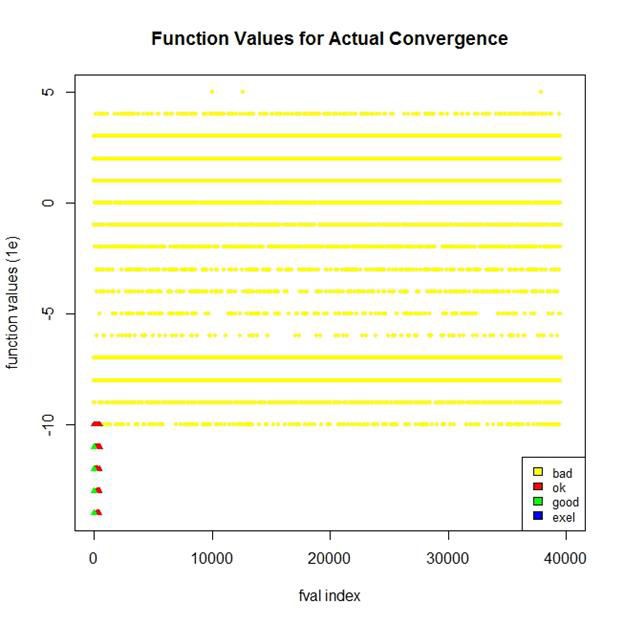
*Graph returns yellow points which do not get converged, while the red points (ok) get converged. We also got some good points (green).*



*The function returned final points in parabolic shape. There are few optimal points near to the real optimal minimum (1,1) while others is far away. The function even returns a point whose the second value is 60000.*

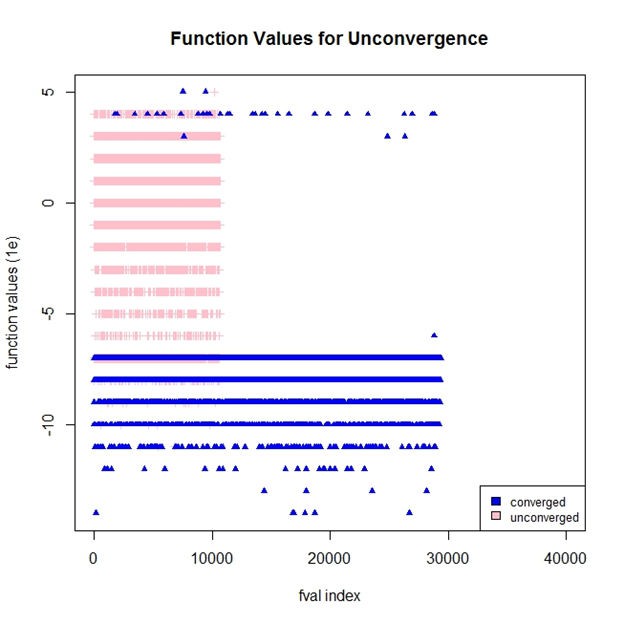
Page | 16

**D. Function value for actual convergence:**



***E. Function Values for un-convergence:***

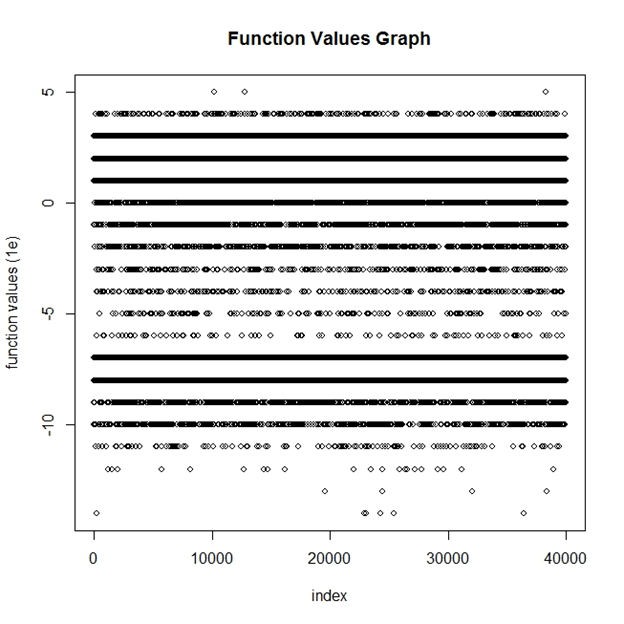
*We have 466 ok points in red color triangle. We also have 8 good points which are marked as green triangles. These point achieved very small FVs (less than 1e-10) which are near to 0.*



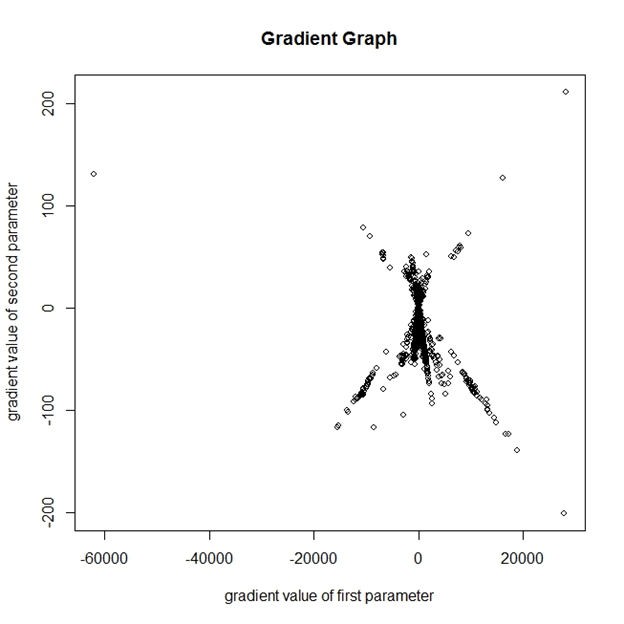
*We can see that there are some pink points which the function claims that those are un-converged points. At those points, optim() will return the non-zero convergence values. Interestingly, there are some converged points whose FVs are quite high (above 1e+3).*

Page | 17

***F. Function Values:***



***G. Gradient Graph:***



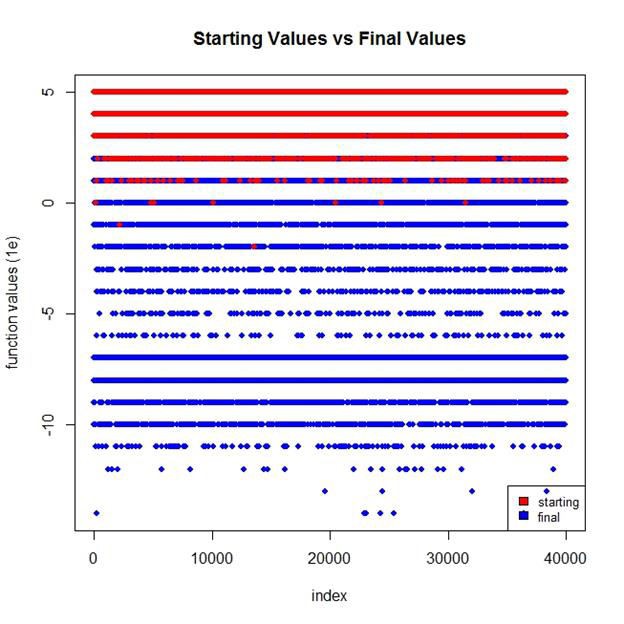
*We can see that most of FVs are lie on lines 1e+1, 1e+2, 1e+3, 1e-7, and*

*1e-8. There are some other points whose FVs attain even below 1e-10.*

*For optimal solution, the gradient should be zero. From the graph, lot of values seem to converge to zero and few points far away from zero.*

Page | 18

***H. Starting Values Vs Final Values:***



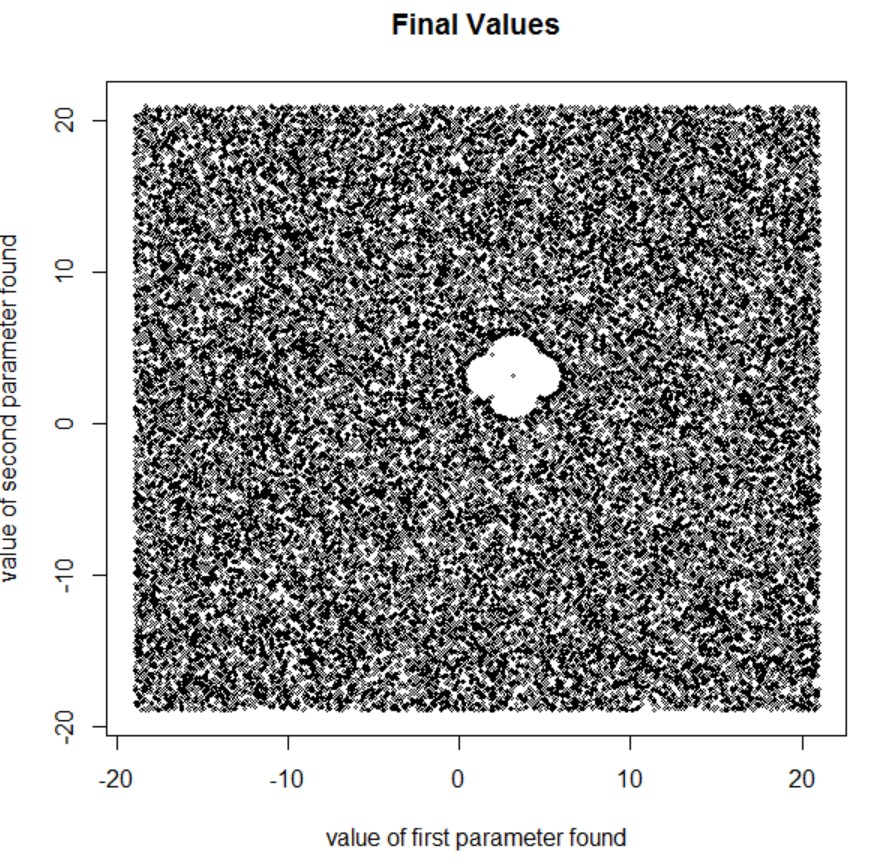
**Method: CG**

**1.Easom function:**

*The final values attain FVs with a very wide range from 1e+3 to 1e-*

*14 while starting values seem to attain FVs with 1e or above.*

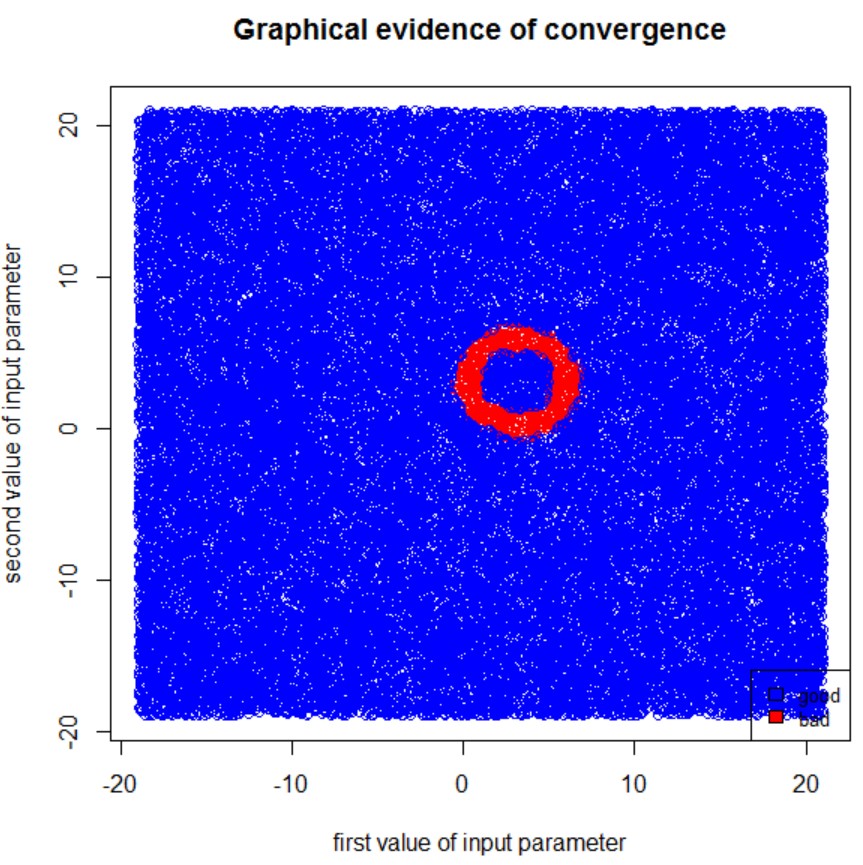
**A. Final Values**



*Final value has one global minima in the center of flowered structure, with a bunch of local minima surrounding it.*

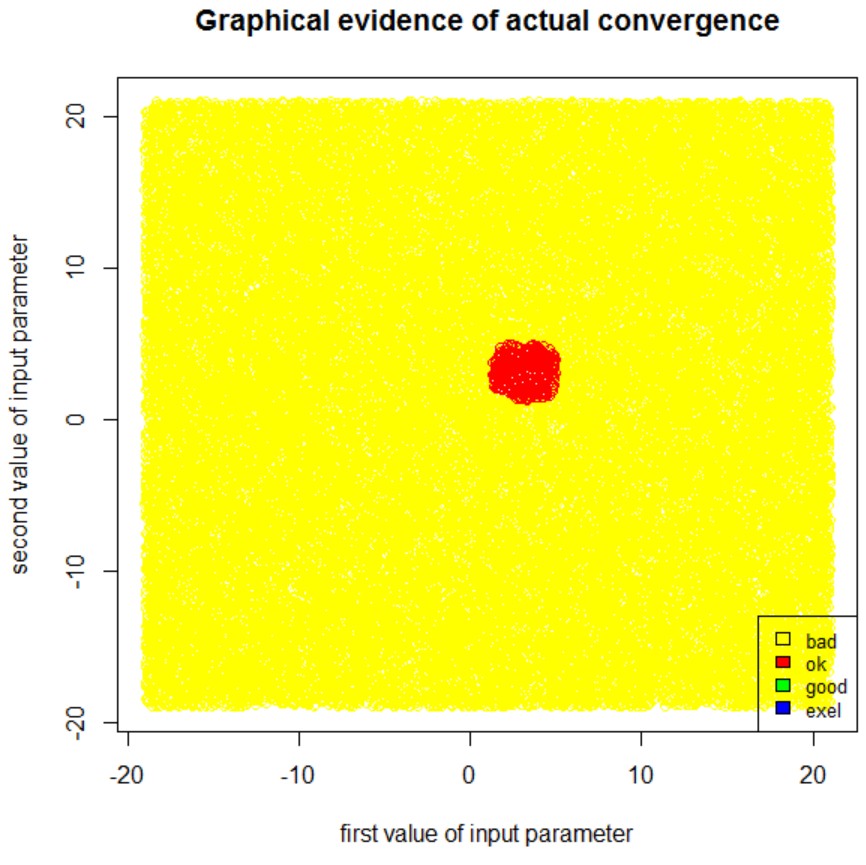
Page | 19

**B. Convergence & Un-convergence:**



*Graph returns red points which doesn’t get converged and is formed around a circular pattern, while the blue points shown in the graph gets fully converged. Most of un-converged pairs of input parameters tend to lie on either x-axis or y-axis and their values seems positive above zero.*

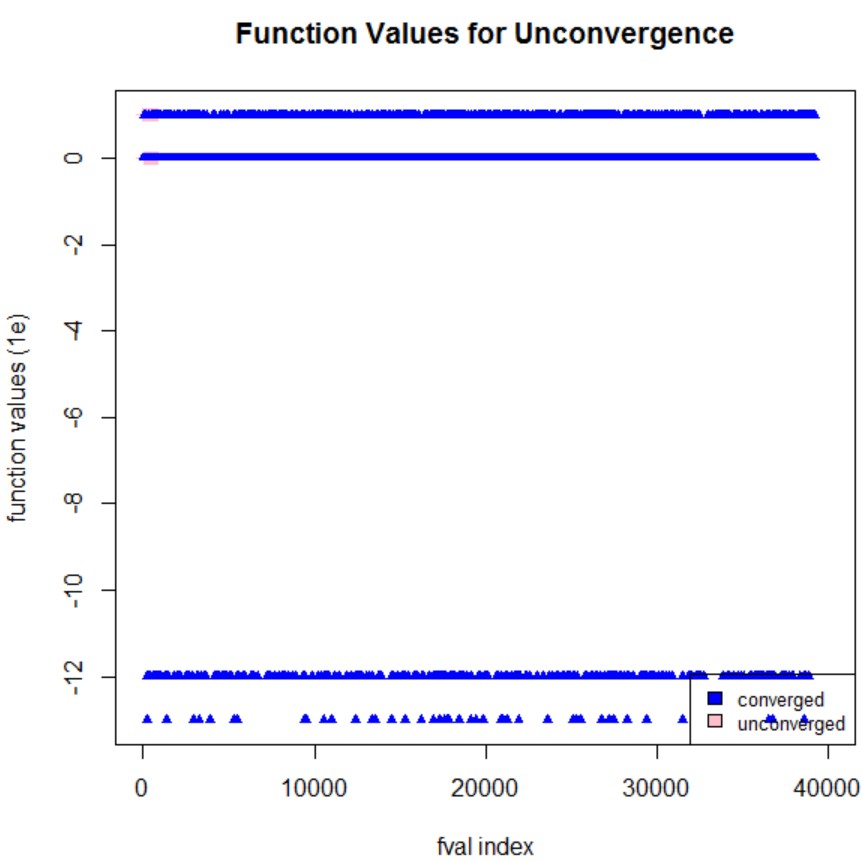
*There are 339 ok points (error norm<1e-5)**that are converged around a circular pattern with a convergence rate of 97.98% and are positive above zero.*



Page | 20

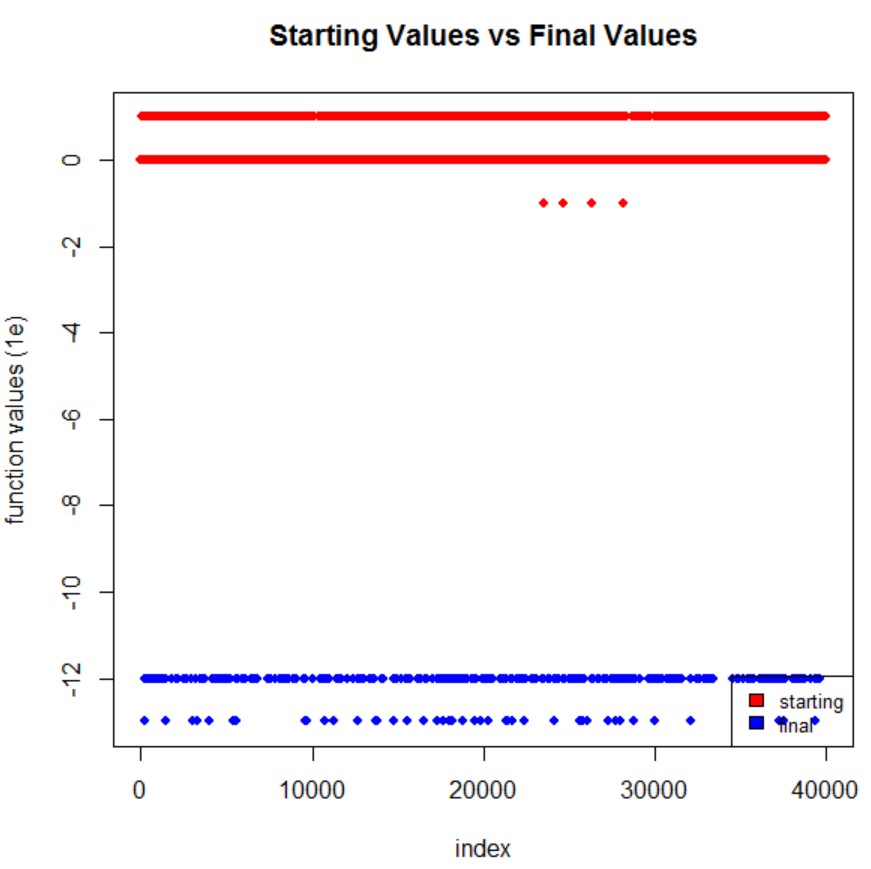


*From the above graph, there are 339 ok points whose function values have actually converged with values less than or equal to* *1e-12. The points shown in red seem to overlap each other and thus appear only as 2 dense points. Also, most of the bad ones are either zero or positive that appears as two strong straight lines in color yellow.*

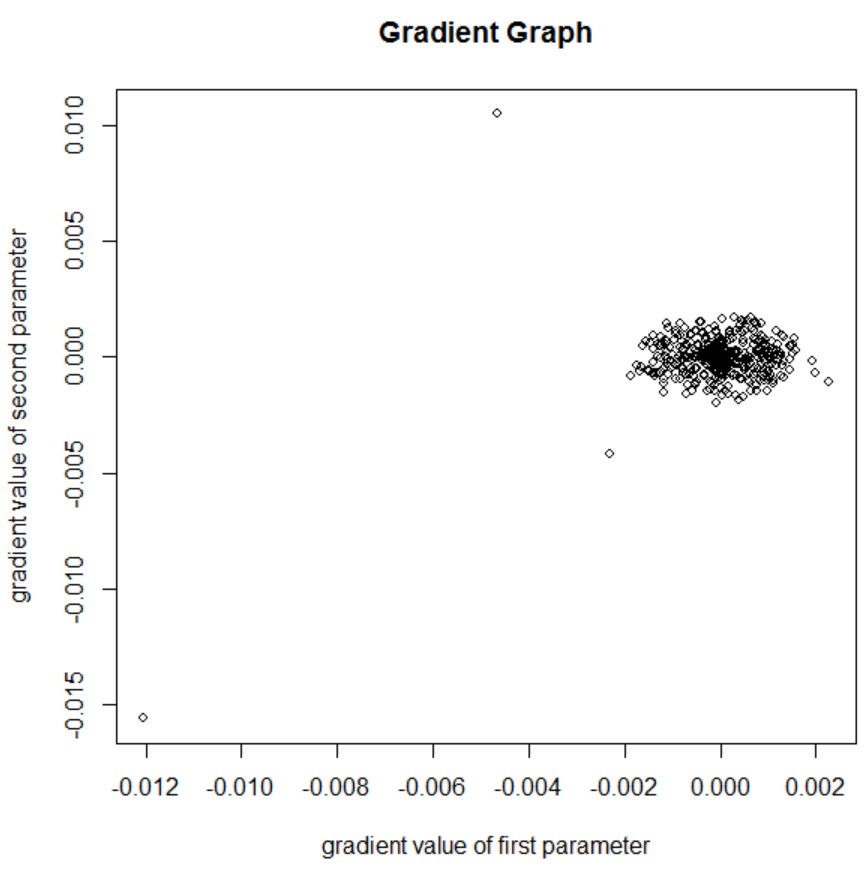


*We can see that there are some pink points on y=0 and above that have been overlapped with blue, for which the function claims that those are un- converged points. At those points, optim() will return the non-zero convergence values.*

Page | 21



**C. Gradient Graph:**



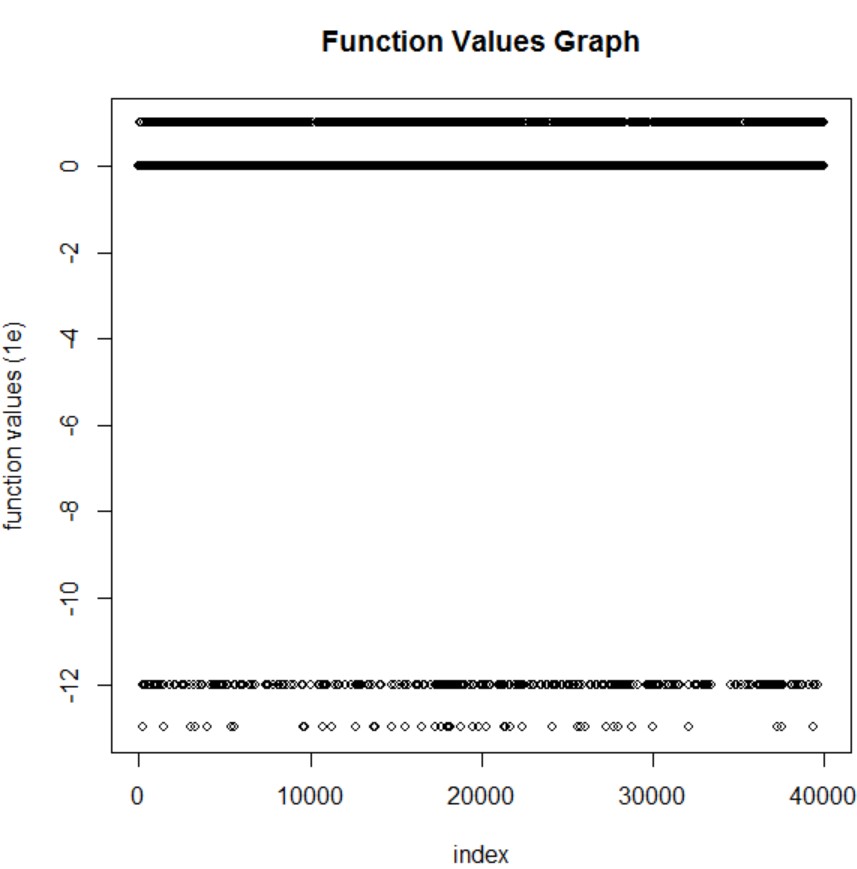
*The final values return a wide range of good function values in color blue around 1e-12 and lesser than that, which is almost equal to zero.*

*For optimal solution, the gradient should be zero. From the graph, lots of values seem to converge to zero and few points clouded near zero. There are couple of points near (-*

*0.012, -0.015) and (-0.004,0.011) that seem to be divergent.*

Page | 22

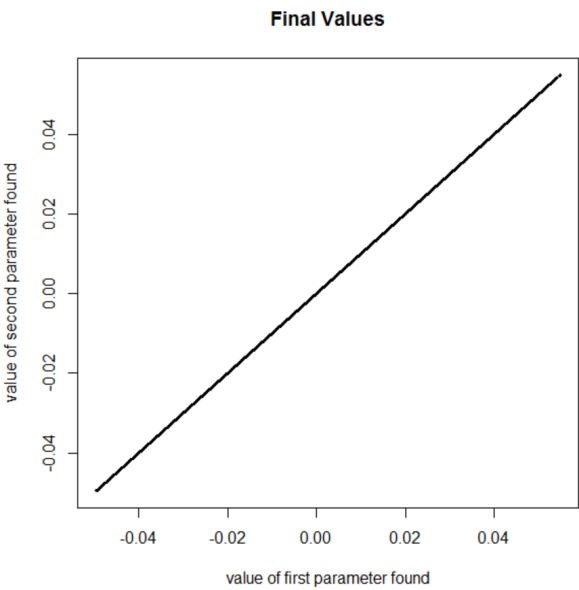
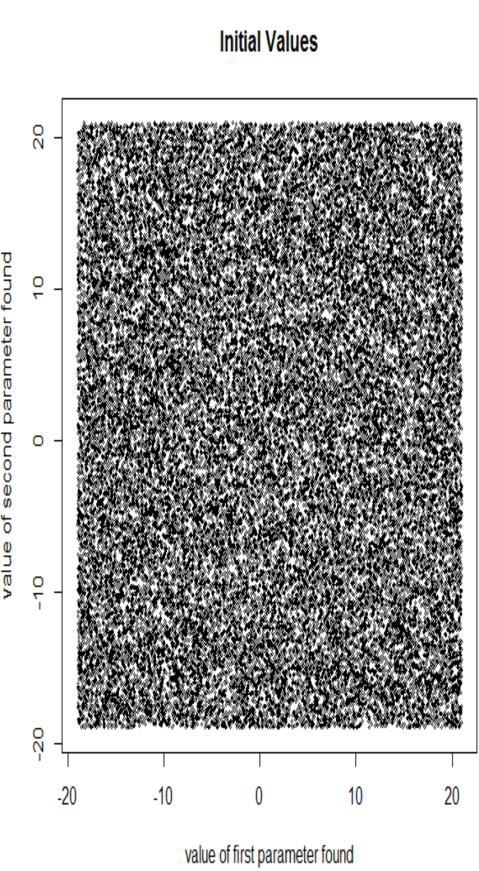
**D. Function Values:**



**2. Matya (CG):**

*There are bad points that lie on y=0 and above. Also, there are few points lie on the line y= -12 and few below that, which are the best values.*

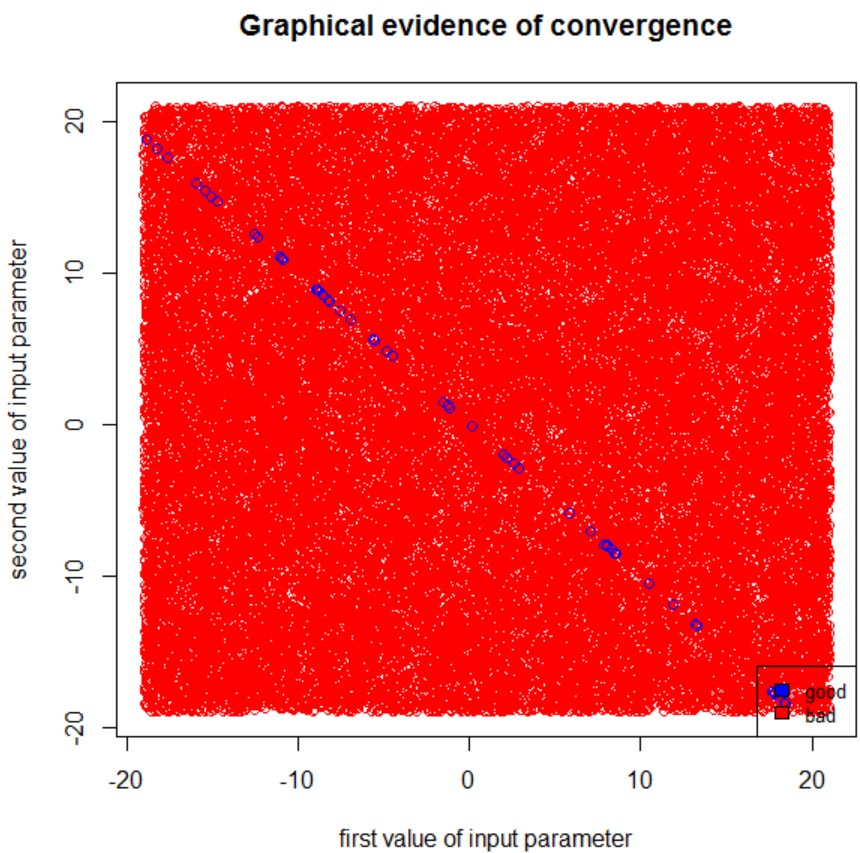
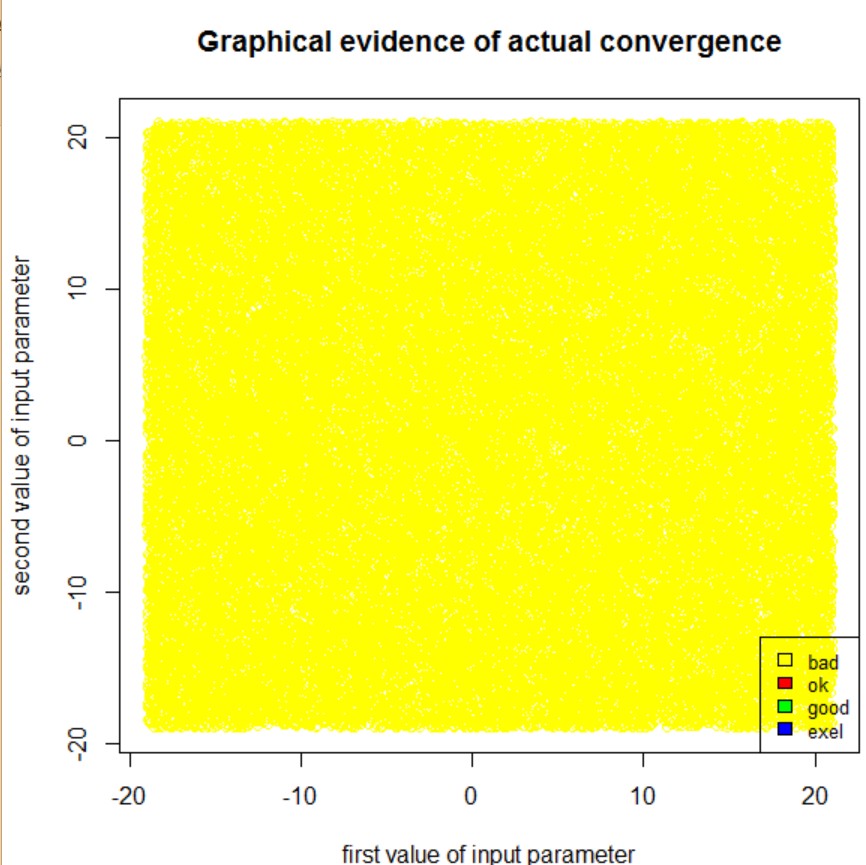
**A. Initial & Final Values:**



*The final value seem to be lie on the diagonal line where the first parameter and the second parameter receive the same value.*

Page | 23

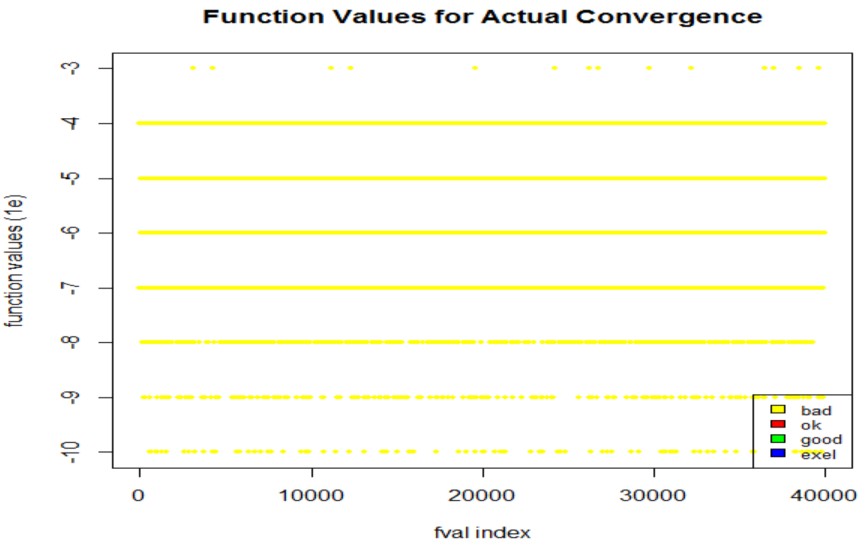
**B. Graphical Evidence of Convergence:**



*As per the graphical evidence of the plot in the left, few good points are converged in color blue diagonally with bad ones scattered throughout the region in color red.*

*The actual convergence graph on the right shows that, there are no points that gets actually converged and all the points of actual convergence are bad shown in color amber.*

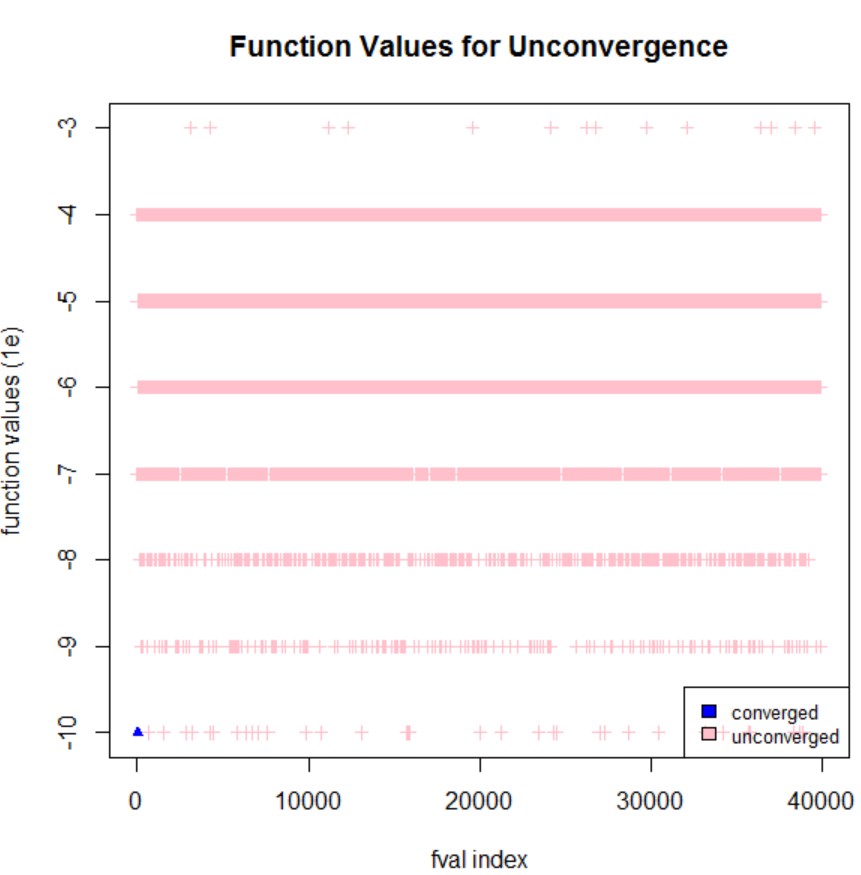
**C. Function Values:**



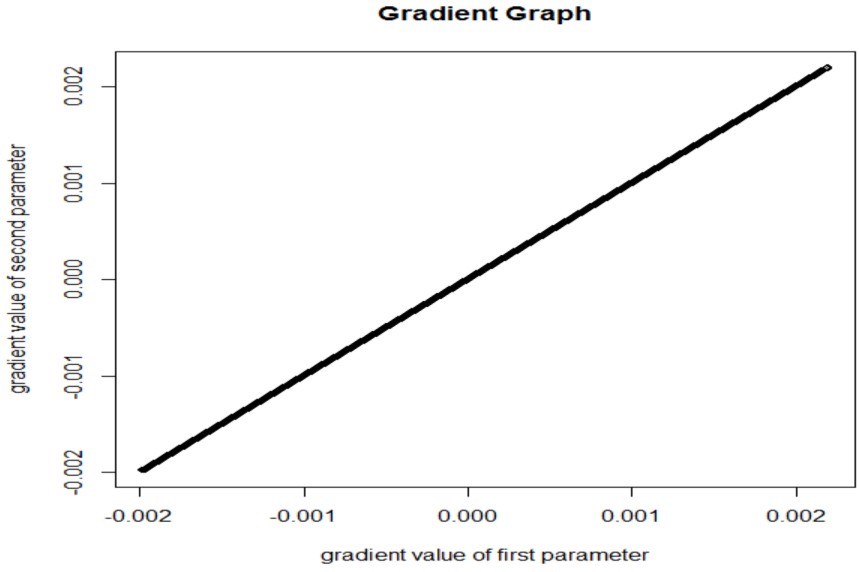
*There are no actual good convergent function value. Therefore, there is no converged point shown in the graph.*

Page | 24

**D. Actual un-convergence:**



**E. Gradient Graph:**



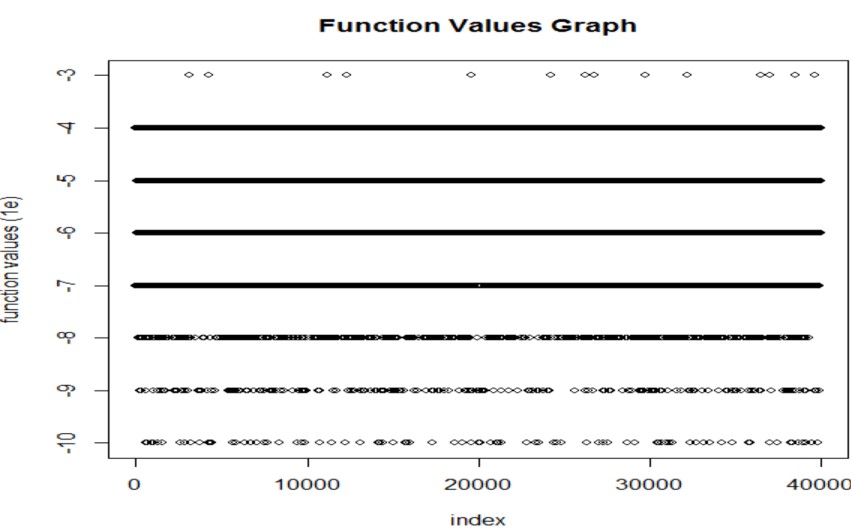
*The actual un-converged function values are in pink and just 1 converged point whose function value in blue along 1e-*

*10.*

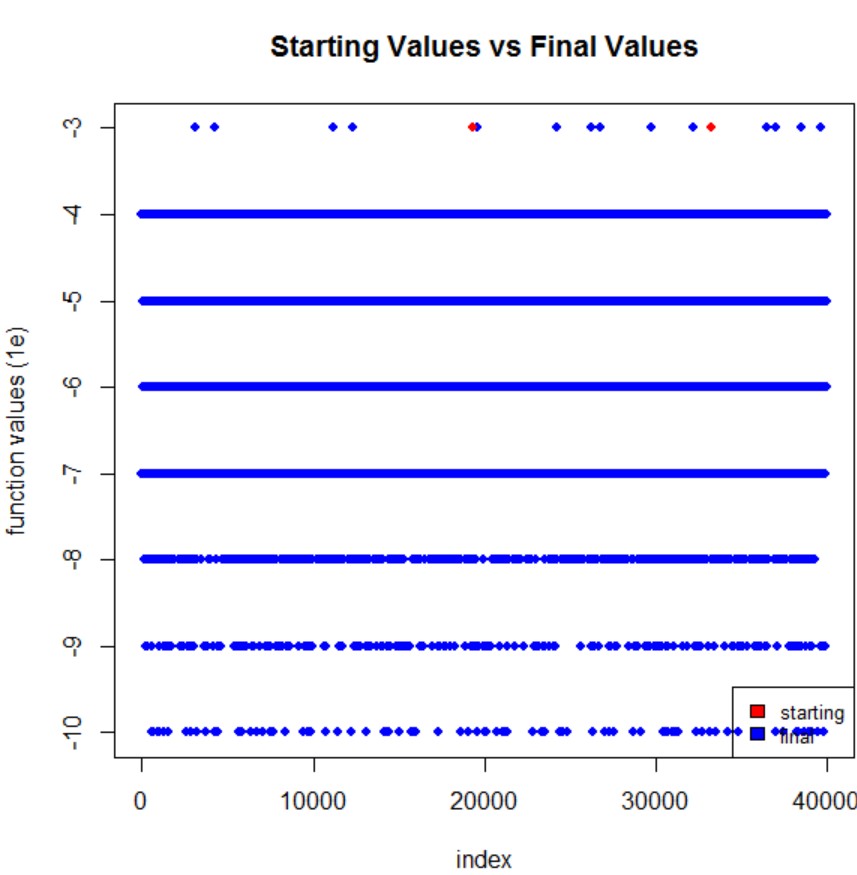
*For optimal solution, the gradient should be zero. From the graph, there are only few values that converged to zero and have dispersed apart diagonally. There are couple of points near (-0.002, -0.002) and (0.002,0.002) that seem to be too divergent.*

Page | 25

**F. Function Values:**



**G. Starting Vs Final Value:**



*There are good converged function values along 1e-10 and below. Those values are sparse.*

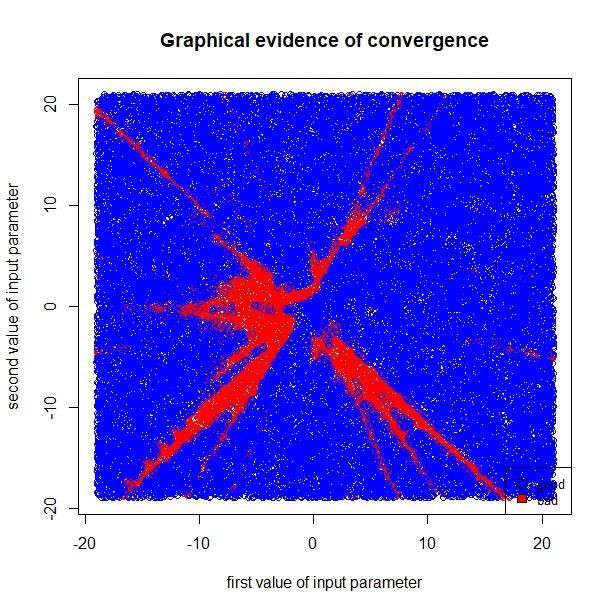
*There are few overlapped starting values in red on y=-3 line which represents the starting values, and final values are dispersed along y=-10 line.*

Page | 26

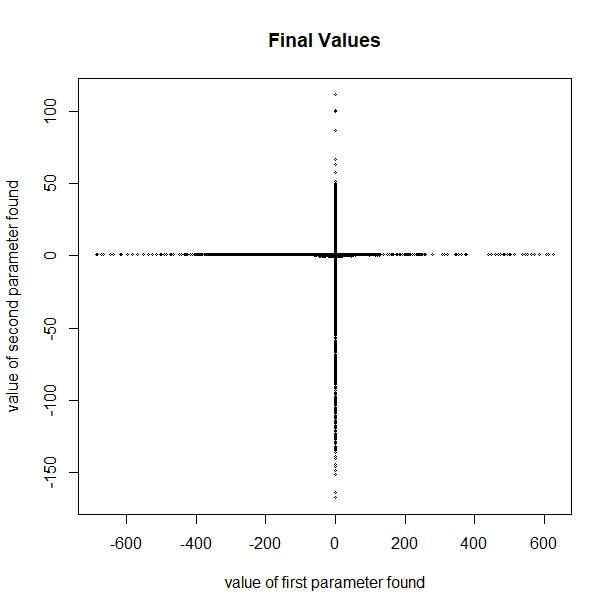
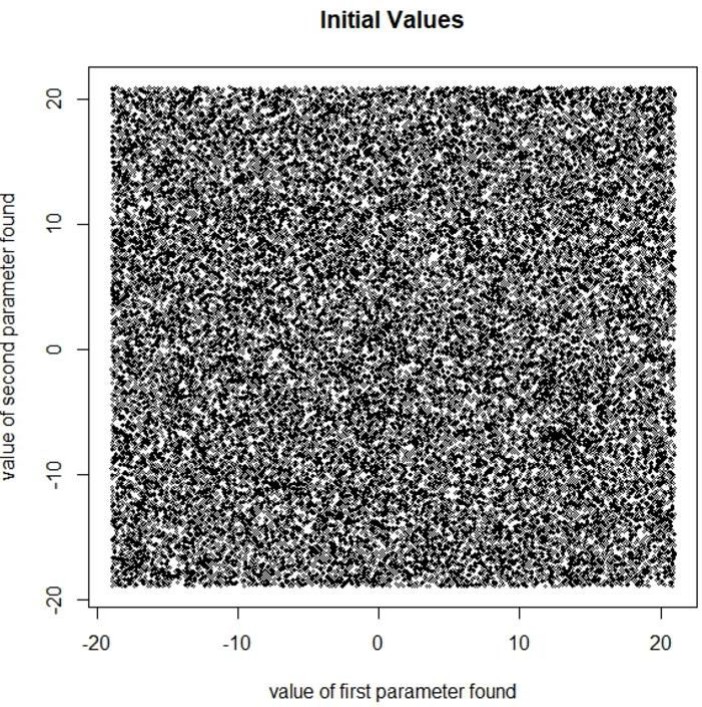
**Optimization using NLM**

**1. Beale Function:**

***A. Evidence of Convergence:***



***B. Initial & Final Values:***



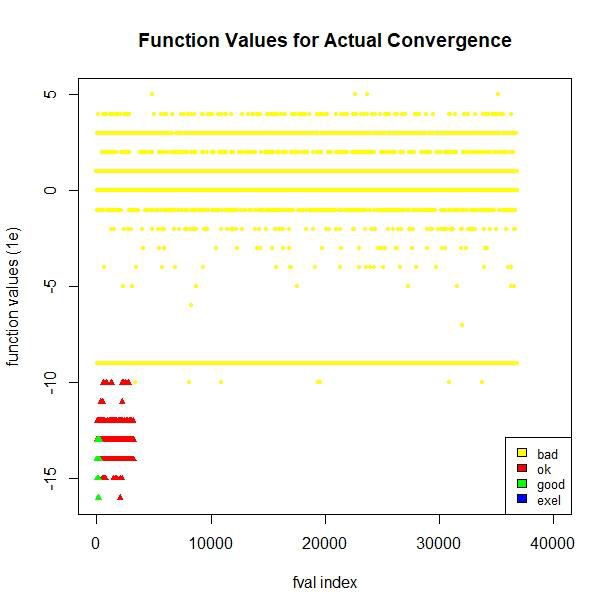
*Graph returns Red points which do not get converged, while the Blue points get fully converged. Most of un-converged pairs of input parameters tend to follow a certain pattern on both diagonals of the graph. They are getting dense when their values is reaching to the Origin.*

*The function returned few good points that are near to the optimal point (3,0.5). Rest of the adjoining point can be considered as slightly converged points. The function also returns several points which are not converged & these points are near (-600,0) and (0,*

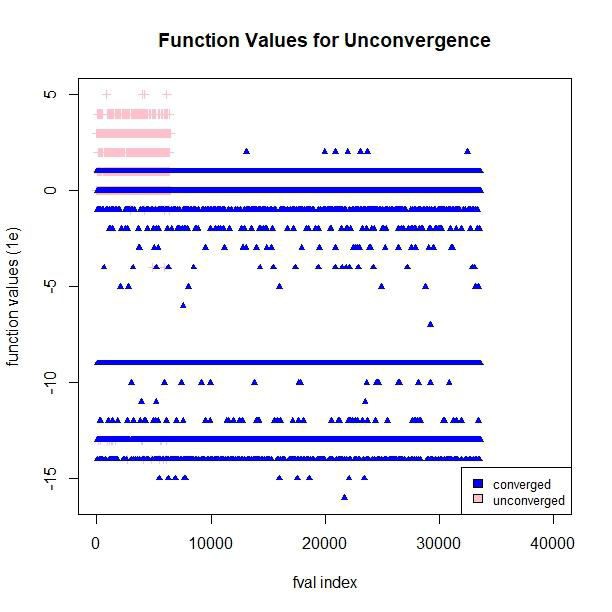
*-150).*

Page | 27

**C. Function value for actual convergence:**



***D. Function Values for un-convergence:***



*We can see from the graph that all of actual ok and actual good points achieved function values*

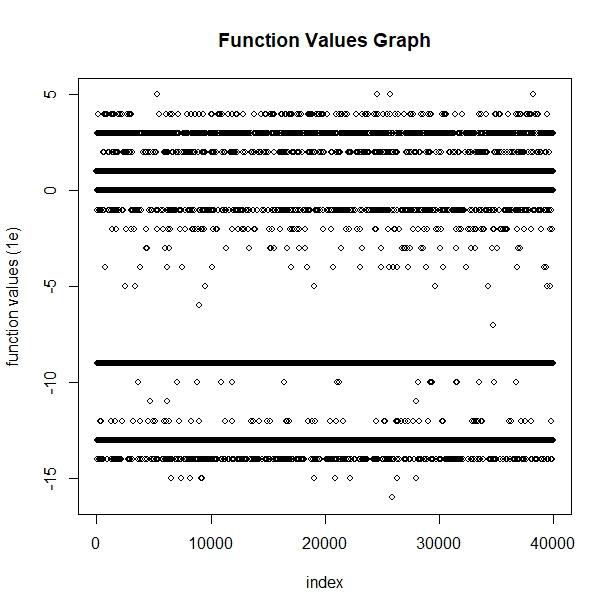
*1e-10 or below. This proves that they are reliable result. Moreover, it is very interesting to be seen that all of them are found at a very early stage (within first 5000 fval-index).*

*There are some pink points which the function claims that those are function value of un-converged points. Theoretically, at those points, nlm() will return the non- zero convergence values. Some of them have function value along 0 line while that of others near to 1e-*

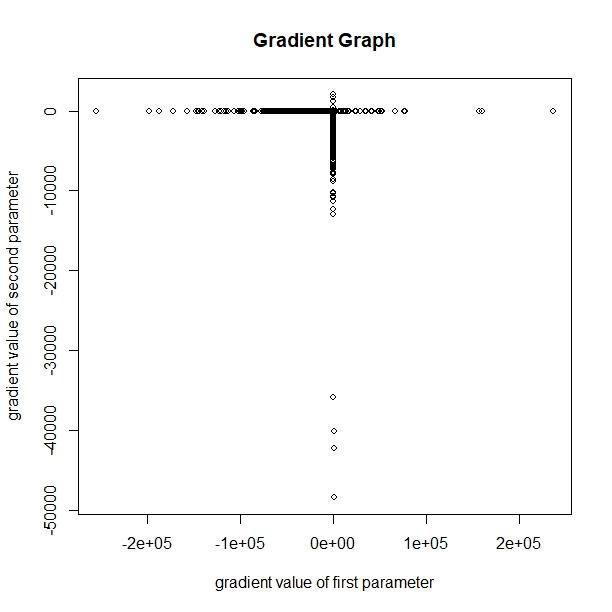
*4*

Page | 28

***E. Function Values:***



***F. Gradient Graph:***



*Most of the time nlm()*

*returns values achieve FV at*

*1e0, 1e+1, 1e-9 and 1e-13. There are some points attain FVs even below the line 1e-*

*13. Clearly, those points may be considered as optimal points.*

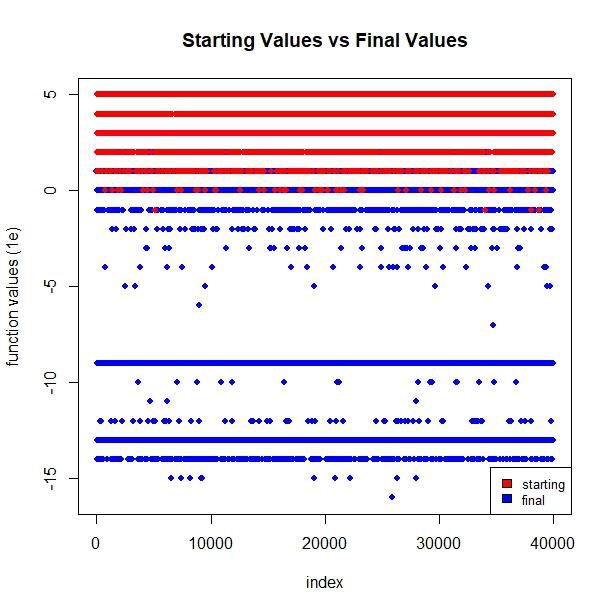
*For optimal solution, the gradient should be zero. From the graph, lot of values seem to converge to zero and few points far away from zero, with values even reach to nearly*

*-50000. Surprisingly, the gradient value for second parameter tend to be negative rather than positive.*

Page | 29

***G. Starting Values Vs Final Values:***

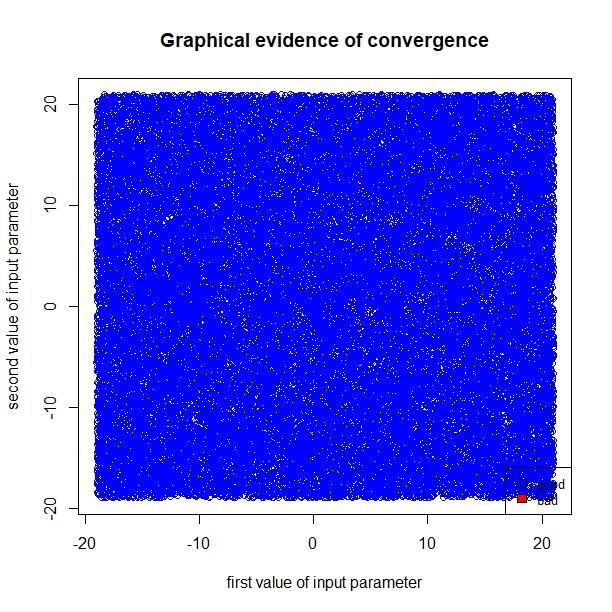
The final values return a wide range of function value. There are some good final values attain function values that are very small (below 1e-15), which are nearly to 0.



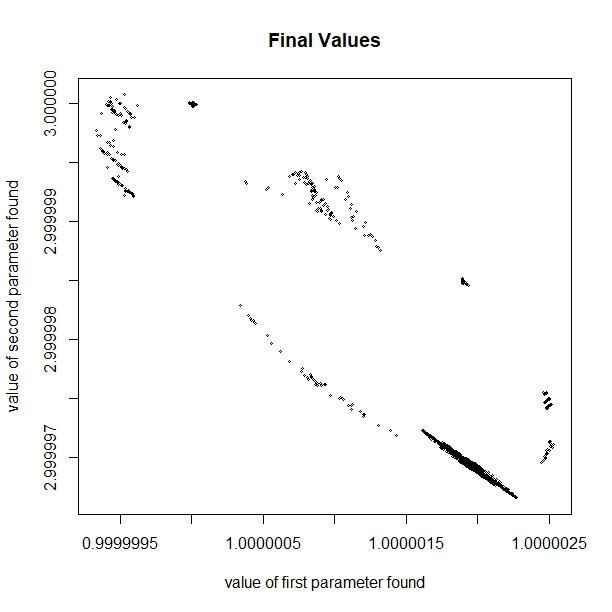
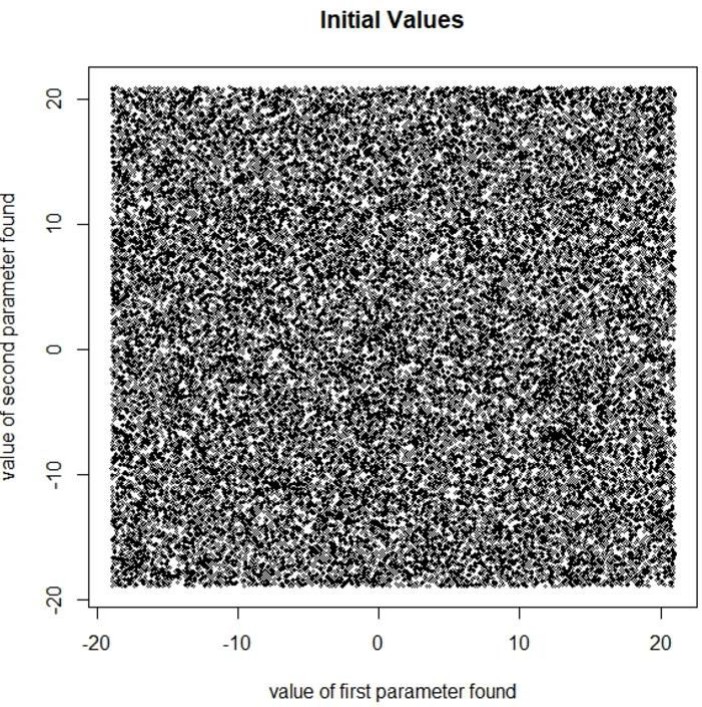
Page | 30

**2. Booth Function (nlm):**

***A. Evidence of Convergence:***



**B. *Initial & Final Values:***



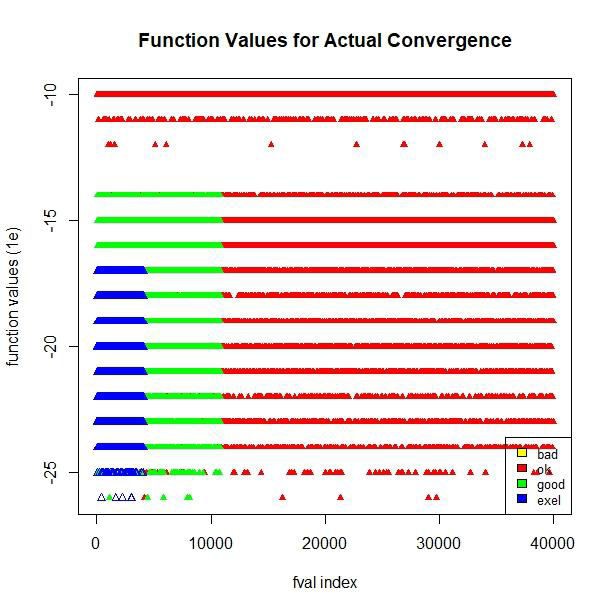
*nlm() claims that 100% of the result are totally converged.*

The function returned few optimal points which are near to (1,3). Most of the time, nlm()

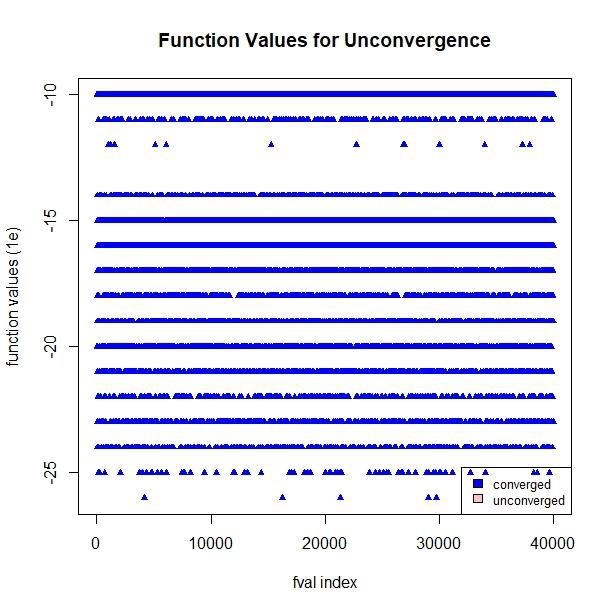
returns values around the area (2.999997, 1.0000020). Clearly, the error between actual optimal minimum point and optimal minimum found are very small around 1e-5

Page | 31

**C. Function value for actual convergence:**



***D. Function Values for un-convergence:***



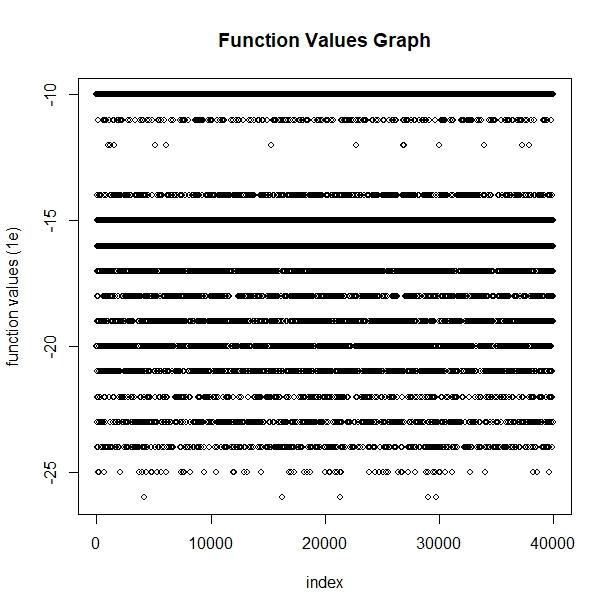
*As nlm() claimed that all of input are converged, their function values are at least ok which are shown in red. For excellent points in blue, their FVs attained at very small values, less than*

*1e-17.*

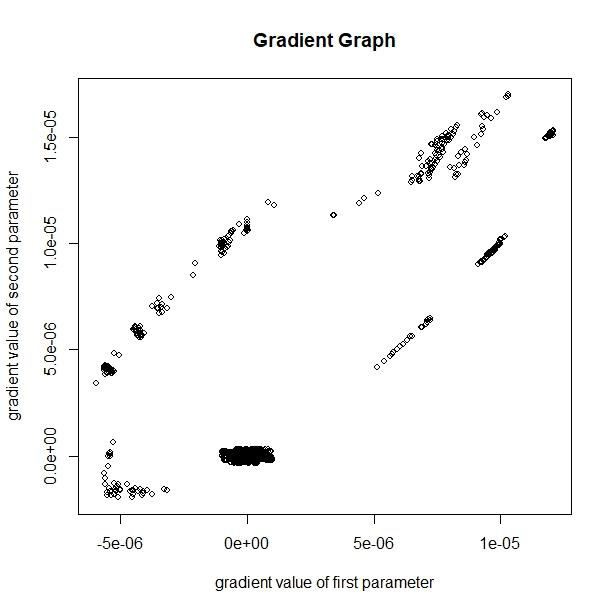
*As nlm() claimed that all input parameter are converged to optimal solution. The number of un-converged points on the graph is 0. Also, we can see that, Converged points achieve FVs at very small. There are few points which their FVs even less than 1e-25*

Page | 32

***E. Function Values:***



***F. Gradient Graph:***



*Most of The FVs are lie on the y=*

*-10 and y=-15 and other FVs tend to get values below that.*

*For optimal solution, the gradient should be zero. From the graph, lot of values seem to converge to zero and few points far away from zero. However, the error between the gradient values of un- converged parameters and 0 are very small.*

Page | 33

***Inference:***

*After testing OPTIM for all the five functions to maximize likelihood using default (Nelder- Mead), BFGS and CG methods,* ***BFGS*** *has good convergence which is a variable metric algorithm that uses function values & gradient to build up a picture of surface to be optimized. Booth and Matya have 100% convergence with both good & excellent values with err.norm<1e-9. While, default method (Nelder-Mead) has very few good points of convergence.*

*On the other hand,* ***NLM*** *performs well with booth function that has 20% excellent points of maximum convergence. While, matya and beale has good convergence rate with more than 90% of decent convergence with err.norm<1e-5.*

*Therefore, the preferred method to maximize likelihood is* ***optim()*** *using* ***BFGS method***

*with* ***control list*** *absolute tolerance of 1e-16.*

Page | 34

***APPENDIX:***

*R code:*

*# function to be changed for every run*

f<-f.beale

actual.optim = **c**(3,0.5)

n = 10

conv = **numeric**(n)

st.x = **runif**(n,-19,21)

st.y = **runif**(n,-19,21)

in.pts = **cbind**(st.x,st.y)

out.pts = **matrix**(**rep**(0,2\*n), ncol = 2)

out.grad = out.pts out.fval = **numeric**(n) ini.fval =**numeric**(n) **set.seed**(2)

for ( i in 1:n )

{

ini.fval[i] = **f**(in.pts[i,])

optim.out = **optim**(in.pts[i,],f, method="BFGS",control=**list**(abstol = 1e-16))

if (optim.out$conv == 0) { conv[i] = 1}

out.pts[i,] = optim.out$par out.fval[i] = optim.out$value

if (out.fval[i]==0){ out.fval[i]=1e-60}

out.grad[i,] = **grad**(f,optim.out$par)

}

*#Graph shows evidence of convergence (q1)*

good = **which**(conv == 1)

bad = **which**(conv == 0)

**dev.new**()

**plot**(in.pts,

main = 'Graphical evidence of convergence',

xlab = 'first value of input parameter', ylab = 'second value of input parameter', type = 'n')

**points**(in.pts[bad,],col = 'red')

**points**(in.pts[good,],col = 'blue')

**legend**('bottomright',**c**('good', 'bad'), cex=0.8, fill=**c**('blue','red'))

*#find the error*

err = out.pts - **matrix**(**rep**(actual.optim,n),ncol = 2, byrow = TRUE)

err.norm = **sqrt**(**rowSums**(err^2))

#### find actual good, bad actual.ok = **which**(err.norm < 1e-5) actual.good = **which**(err.norm < 1e-7) actual.excel = **which**(err.norm < 1e-9)

Page | 35

actual.bad = **which**(err.norm >= 1e-5)

**length**(actual.ok)

**length**(actual.good)

**length**(actual.excel)

**length**(actual.bad)

*# #Graph shows evidence of actual convergence*

**dev.new**()

**plot**(in.pts,

main = 'Graphical evidence of actual convergence', xlab = 'first value of input parameter',

ylab = 'second value of input parameter', type = 'n')

**points**(in.pts[actual.bad,], col = 'yellow')

**points**(in.pts[actual.ok,], col = 'red') **points**(in.pts[actual.good,], col = 'green',pch = 19) **points**(in.pts[actual.excel,], col = 'blue',pch = 17) **legend**('bottomright',**c**('bad','ok','good','exel'), cex=0.8, fill=**c**('yellow','r ed','green','blue'))

grad.norm = **sqrt**(**rowSums**(out.grad^2))

grad.tol.1 = **length**(**which**(grad.norm < 1e-6))

*# Graph shows the function values when actual convergence happens (q4)*

**dev.new**();**plot**(**ceiling**(**log10**(out.fval)), col='white',cex=0.8,pch=17,

main = 'Function Values for Actual Convergence',

xlab = 'fval index',

ylab = 'function values (1e)')

**points**(**ceiling**(**log10**(out.fval[actual.bad])),col = 'yellow', cex=0.8,pch = 20)

**points**(**ceiling**(**log10**(out.fval[actual.ok])), col = 'red', cex=0.8,pch = 17)

**points**(**ceiling**(**log10**(out.fval[actual.good])), col = 'green', cex=0.8,pch = 17

)

**points**(**ceiling**(**log10**(out.fval[actual.excel])), col = 'blue', cex=0.8,pch = 24

)

**legend**('bottomright',**c**('bad','ok','good','exel'), cex=0.8, fill=**c**('yellow','r

ed','green','blue'))

###############new graph#####################

*# Graph shows the function values when the function falsely claimed to achiev e convergence (q5)*

**dev.new**();**plot**(**ceiling**(**log10**(out.fval)), col='white',cex=0.8,pch=17, main = 'Function Values for Unconvergence',

xlab = 'fval index',

ylab = 'function values (1e)') **points**(**ceiling**(**log10**(out.fval[bad])), col='pink',cex=0.8,pch=3) **points**(**ceiling**(**log10**(out.fval[good])), col = 'blue',cex=0.8,pch = 17) **legend**('bottomright',**c**('converged','unconverged'), cex=0.8, fill=**c**('blue','pi

Page | 36

nk'))

*# In.pts, out.pts, grad, fval graphs (q6)*

**dev.new**();**plot**(in.pts,

main = 'Initial Values',

xlab = 'value of first parameter found', ylab = 'value of second parameter found', cex=0.4)

**dev.new**();**plot**(out.pts,

main = 'Final Values',

xlab = 'value of first parameter found',

ylab = 'value of second parameter found', cex = 0.4)

**dev.new**();**plot**(out.grad,

main = 'Gradient Graph',

xlab = 'gradient value of first parameter', ylab = 'gradient value of second parameter',

cex = 0.8)

**dev.new**();**plot**(**ceiling**(**log10**(out.fval)),

main = 'Function Values Graph',

xlab = 'index',

ylab = 'function values (1e)', cex=0.8)

*#Graph shows a plot with starting values and final values*

**dev.new**();**plot**(**ceiling**(**log10**(out.fval)),col = 'blue',cex=0.8,pch = 19,

main = 'Starting Values vs Final Values', xlab = 'index',

ylab = 'function values (1e)')

**points**(**ceiling**(**log10**(ini.fval)), col = 'red',cex=0.8,pch = 19)

**legend**('bottomright',**c**('starting','final'), cex=0.8, fill=**c**('red','blue'))

*#Proportion of time convergence was reached*

**paste**(**length**(**which**(conv == 1))\*100/n,'%', sep="")

*#Time taken to converge*

**length**(**which**(conv == 1))

Page | 37