Task 1 Report

Tables of Obtained Results

	Run 1			Run 2		
p = 80 & z = 0.1	# of sol searched	sol	f(sol)	# of sol searched	sol	f(sol)
(2,2)	2560	[2.9758579 0.49315102]	0.0001101863136	2400	[2.97801044 0.49621372]	0.0001454191537
(1,4)	5440	[-3.28193836 1.2421894]	0.8632833269	4320	[-2.72855509 1.28319411]	0.9343056093
(-2,-3)	4560	[2.99442251 0.49827829]	7.58E-06	4480	[3.02502465 0.50535645]	0.0001121827867
(1,-2)	2800	[2.95112499 0.4854259]	0.000508591256	2880	[2.96766375 0.49287045]	0.0001954731496

	Run 1			Run 2		
p = 80 & z = 0.02	# of sol searched	sol	f(sol)	# of sol searched	sol	f(sol)
(2,2)	12720	[2.9961921 0.499417]	5.35E-06	12480	[3.00542234 0.5011113]	5.91E-06
(1,4)	24800	[-3.49606901 1.23124694]	0.8408570859	22800	[-3.12336819 1.2539362]	0.8813098312
(-2,-3)	22720	[3.00277838 0.50042732]	2.81E-06	22480	[3.00428829 0.50147903]	6.98E-06
(1,-2)	14080	[3.00118402 0.50045307]	8.13E-07	14160	[2.99599153 0.49879781]	3.56E-06

	Run 1			Run 2		
p = 500 & z = 0.1	# of sol searched	sol	f(sol)	# of sol searched	sol	f(sol)
(2,2)	16000	[3.0039086 0.50159518]	1.16E-05	15500	[3.00835348 0.50349248]	5.88E-05
(1,4)	59000	[-8.38650649 1.10677165]	0.6260490894	43500	[-5.95868577 1.14620223]	0.6915511253
(-2,-3)	27000	[3.00195164 0.49963834]	1.71E-05	26500	[2.99767562 0.49990877]	6.29E-06
(1,-2)	17000	[3.00951154	1.96E-05	17000	[3.00276342	1.51E-05

		0.50282191]			0.49990897]	
	Run 1			Run 2		
p = 500 & z = 0.02	# of sol searched	sol	f(sol)	# of sol searched	sol	f(sol)
(2,2)	76500	[2.99833947 0.49950841]	5.88E-07	75500	[3.00073987 0.50018716]	8.79E-08
(1,4)	305500	[-9.02714759 1.10038805]	0.6138578736	315500	[-9.30503845 1.09765413]	0.6092414495
(-2,-3)	131000	[2.99915152 0.49961073]	8.52E-07	131000	[2.99915444 0.49994518]	6.68E-07
(1,-2)	81000	[3.00143674 0.50024857]	5.97E-07	82000	[2.99966961 0.49989166]	3.36E-08
	33rd Run					
p = 50 & z = 1e-08	# of sol searched	sol	f(sol)			
[2.99966961, 0.49989166]	1871250	[3. 0.5]	2.68E-18			

Seed Values Used

Run 1 seed = 42
Run 2 seed = 77977

Expected Results Interpretation

Solutions (x,y) generated tend to be close to (3.0, 0.5). Overall, I judge the quality of the solutions generated by RHC as medium. For the 32 runs summarized in the 4 tables above, the maximum (worst) f(x,y) = 0.9343056093; the minimum (best) f(x,y) = 3.36E-08. My implementation of RHC tends to produce f(x,y) values that are closer to the minimum 3.36E-08, but when RHC performs poorly, we see that the results are quite bad. 0.9343056093 is relatively large considering the true minimum of f(x,y) = 0. We see from the table summary that the best f(x,y) values are generated with p = 500 and z = 0.02. In other words, the best f(x,y) values are found in the last table because here p is large and z is small compared to the other tables. With this understanding it follows that the worst f(x,y) values should be found in the first table since p is small and z is large compared to the other tables. Observe that this is the case in the table summary above.

Starting point (2,2) is closest to the true solution (3.0, 0.5), so it makes sense that for all runs with starting point (2,2) less solutions are searched compared to other starting points. Note that, for a given starting point, number of solutions searched appears directly proportional to p/z. That is, number of solutions searched tends to increase as p increases and number of solutions searched tends to increase as z decreases (holding starting point fixed).

My implementation of RHC takes advantage of SIMD capabilities of NumPy arrays. For example, after generating the neighborhood array the function to minimize is applied to each element of the neighborhood (an (x,y) coordinate). NumPy applies this function element-wise as a vector operation. So, where possible, NumPy parallelizes these operations. The result of this type of implementation is faster algorithm speed compared to using Python lists.

Summary of 33rd Run

I chose the best (x,y) generated from the previous 32 runs as the starting point for my 33rd run. Next, I decided that a relatively larger p value and relatively smaller z value would likely increase the quality of f(x,y) generated. From the table summary above, we see that I obtained the best results overall in the 33rd run of RHC. The last run generated (x,y) = (3.0, 0.5) and f(x,y) = 2.68E-18. I rate the quality of this solution as good because if we were to represent 2.68E-18 with single precision (assuming 4 byte word) the value would round to zero which is the true solution. If we were interested in double precision for our solution we could (keeping the seed and starting point the same) simply increase p and decrease z. This will increase the run time of the program, but will produce more accurate results.