Optimization Mini-Project
Enhancing the simple hill-climber

PBA Software development 2020

Lars Søndergaard Petersen

Table of content

1. Your experiments in general.

In my experiments I have looped all the suggested step size, iteration and neighbour combinations for each problem resulting in twenty-seven combinations per problem.

In appendix 1 I have noted down the parameters, number of calls to eval, the best point found and ordered the experiments by the highest evaluated value, this data can also be found in the excel file. All files and code can also be found on GitHub: https://github.com/trezum/OptimeringMiniProject

P1 has a very simple landscape, it only one hill that always leads to the global maxima. Therefore, as the experiments show, we get a very precise evaluation value with even the smallest number of eval calls. The worst value I got was 0,99999502046902, so 5 decimals of precision in the unluckiest case.

P2 also has a simple landscape but instead of just having one hill it has two where one contains a local optimum and the other contains the global maxima. Because of the local optimum this problem takes a bit more effort/luck to get the same 5 decimal precision. Five out of the twenty-seven experiments did not achieve 5 decimals of precision.

RevAckley on the other hand has a much more complex landscape with lots of local optimums thereby making it hard to find/luck out on finding the global optima. None of my experiments reached 5 decimal precision, but two of them did reach 4 decimals.

2. Where do you think improvement happens in the iterated hill – climber?

The ways our implementation of the iterated hill climber is improved compared to the first hill-climber we made are clear.

- By iterating the hill-climber we give it more starting points, there by a higher chance of starting out on the correct hill at least once.
- In the other hill climber, the step size was always 1 step in both directions in the array. By introducing step size and adding a degree of randomness to the step selection and step size we get a chance of selecting a better neighbour than before when it was hardcoded.
- We also select more neighbours, before we only looked at two, now it is a parameter and we look at 10, 100 and 200 in the experiments. This gives a higher chance of selecting a neighbour.

3. The effectiveness of the experiments concerning the number of new solutions created.

The numbers here are all form P1

Varying step size			
Iterations	Step size	Neighbours	Calls to eval
100	0.01	100	1437634
100	0.001	100	13877905
100	0.1	100	166145
Varying number of neighbours			
Iterations	Step size	Neighbours	Calls to eval
100	0.01	200	2821538
100	0.01	100	1437634
100	0.01	10	201170
Varying iterations			
Iterations	Step size	Neighbours	Calls to eval
100	0.01	100	1437634
200	0.01	100	3014647
10	0.01	100	140906

a. Can you explain the numbers?
   - More iterations add more eval calls, because the whole hill-climb is run more times.
   - Smaller step size adds more eval calls because it takes more steps for the hill-climber to get to a maximum.
   - Adding more neighbours also increase the number of eval calls because we need to compare more neighbours to each other. To do this we need to evaluate them.

4. Based on your experiments, what do you think the global optima for the functions are?

P1 experimentation shows the global optima to very likely be at the point 0,0 with the evaluated value 1. Evaluating this point in code gives the expected output of 1.

P2 on the other hand is not so straightforward because the experiments show that the global optima is at some fraction without repeating decimal points. Therefor my best guess is the best found point at [1.6973296405794807, 1.697333243601135], evaluating to 2,00311674604800.

RevAckley has a global optimum witch is, like P1 a, whole number. There for when the hill-climber gets close enough to the global optima, we start to get repeating decimal points. My expectation is that the point that evaluates to 0 will be the global optima. Evaluating the point 0,0 in code gives us the value of 0.

5. Give both the best solutions found and the evaluation function value of these.

	Eval	Point
P1	0,99999999999252	[-2.7351094690310554E-6, 6.04746641305338E-8]
P2	2,00311674604800	[1.6973296405794807, 1.697333243601135]
RevAckley	-0,00003440397255	[-9.615795108470324E-6, 7.446930512685743E-6]

6. How close do you think you came to the global optima? Can you find them analytically?

From the experimentation the found values seem to be very close to the global optima. The guessed points 0,0 for P1 and 0,0 for RevAckley gives very good evaluations and I am very confident the global optima has been found.

As for P2 I suspect the precision to be like the one from P1 probably slightly lower based on the observation made in the introduction, P2 having less experiments with 5 decimal points of precision.

The best point found for P2 does seem to have some symmetry in the decimals, if this is true one of the coordinates is probably better than the other.:

	Eval	Point
Found	2.003116746048002	[1.6973296405794807, 1.697333243601135]
Repeated left	2.0031167460587516	[1.6973296405794807, 1.6973296405794807]
Repeated Right	2.003116746037723	[1.697333243601135, 1.697333243601135]

Repeating the left coordinate gives a better evaluation.

If a more precise maximum is needed, I can start the hillclimber out with the known best point instead of randomly, by doing this we can increase the amount of exploitation done by the algorithm. I would also decrease the step size.

Doing so yielded the following maximum:

Eval: 2.003116746063169

Point: [1.6973307014531214, 1.6973307025972542]

Adding some symmetry to the neighbour selection could improve the hillclimber for P2 specifically but would of course, like the other experimentations done here, make it unable to generalize for other problems.

The math for maximizing P2 is beyond me. Wolfram alpha did not give any maximum either but it does have some good visualization graph of the equation.

7. What was the best parameter setting for your iterated hill – climber?

The best parameter settings depend on the problem and on the precision required for whatever application is being working on.

As for the three problems here I can go as precise as needed, limited by the double data type, because the evaluation function does not have any uncertainty. With the goal of three decimals of precision, settings like the ones in the table below should be sufficient while also keeping the calls to eval relatively low:

	Iterations	Step size	Neighbours
P1	10	0.1	10
P2	10	0.1	100
RevAckley	200	0.001	10

Appendix 1

Problem P1

Iterations	Stepsize	Neighbours	Calls to eval	Evaluated value	Best point found
200	0.001	100	28230711	0,99999999999252	[-2.7351094690310554E-6, 6.04746641305338E-8]
100	0.01	100	1437634	0,99999999997594	[4.883247072800872E-6, -4.579790927832883E-7]
200	0.001	200	55933074	0,99999999993684	[-7.625043026046304E-7, -7.91082574333414E-6]
100	0.001	200	29194547	0,99999999993455	[4.625367776464303E-6, -6.637587940342839E-6]
100	0.001	100	13877905	0,99999999992997	[-3.028702953327023E-6, 7.800994393132913E-6]
10	0.001	100	1519657	0,99999999973493	[8.302325685234698E-6, 1.4005023103164048E-5]
10	0.001	200	3286160	0,99999999965823	[1.1100277736268193E-5, -1.4783423095697495E-5]
200	0.001	10	1751236	0,99999999852607	[1.9279639196295153E-5, 3.319981271742014E-5]
200	0.01	100	3014647	0,99999999802169	[-4.419814152746939E-5, 4.9829109621875745E-6]
100	0.001	10	878220	0,99999999800931	[-4.4586316551229236E-5, 1.659055114143818E-6]
100	0.01	200	2821538	0,99999999608239	[2.812086589878797E-5, -5.591801986538762E-5]
10	0.001	10	79838	0,99999999509747	[1.8304229042959803E-5, 6.758317182001149E-5]
200	0.01	200	5600664	0,99999999469048	[5.106829461357847E-5, -5.197646009393942E-5]
100	0.01	10	201170	0,99999999129666	[9.190418532148675E-5, -1.6030124397238598E-5]
200	0.01	10	378535	0,99999998372205	[-1.2274981402500638E-4, 3.4791264873626666E-5]
10	0.01	200	309752	0,99999997037607	[-5.806211327752733E-6, 1.7201808745466083E-4]
200	0.1	200	628527	0,99999996575414	[-1.7644073664169983E-4, 5.5807933496536746E-5]
100	0.1	200	322304	0,99999996502212	[1.1613901345741695E-5, -1.8666279267750586E-4]
10	0.01	100	140906	0,99999990747328	[2.6955697373981687E-4, 1.4094593876207221E-4]
200	0.1	100	343601	0,99999986560163	[-2.793241066749892E-4, -2.3743718862288214E-4]
100	0.1	100	166145	0,99999983255751	[1.4477375171508683E-4, -3.827310667940814E-4]
10	0.01	10	19734	0,99999979775524	[-3.9271500689347935E-4, 2.1913397871280583E-4]
10	0.1	10	2068	0,99999851386049	[-5.91658209936341E-5, -0.0012176370652973861]
10	0.1	200	34181	0,99999788562566	[-0.0014240047670180376, -2.942566870022269E-4]
100	0.1	10	21320	0,99999737742150	[0.0016188619284674445, 4.322036651488448E-5]
200	0.1	10	45653	0,99999708943261	[-6.297091743912916E-4, 0.0015855718163346952]
10	0.1	100	17787	0,99999502046902	[-3.2191218207226757E-4, -0.0022081476222515425]

Problem P2

Iterations	Stepsize	Neighbours	Calls to eval	Evaluated value	Best point found
200	0.001	200	52994655	2,00311674604800	[1.6973296405794807, 1.697333243601135]
10	0.001	200	2160359	2,00311674603738	[1.6973289941630245, 1.6973338595668233]
100	0.001	10	772565	2,00311674584644	[1.697320485858685, 1.6973328108946777]
200	0.001	100	25127486	2,00311674583307	[1.697326234933026, 1.6973404564619328]
100	0.001	100	12392195	2,00311674581196	[1.697337986257137, 1.6973393383045203]
100	0.001	200	26541146	2,00311674563750	[1.6973406326270741, 1.6973200299603364]
100	0.01	10	159271	2,00311674543986	[1.6973181868468614, 1.6973431334183444]
100	0.01	200	2663954	2,00311674441227	[1.6973468400739053, 1.6973069475296356]
10	0.001	100	1470369	2,00311674367593	[1.6973093910763732, 1.6973578662340394]
200	0.01	200	5343585	2,00311674265884	[1.6973289534405651, 1.697372060641609]
200	0.001	10	1652610	2,00311674168318	[1.6973504338781842, 1.6972882521318122]
100	0.01	100	1181296	2,00311669019361	[1.6974037409412266, 1.6971803321764927]
200	0.01	100	2617818	2,00311668793000	[1.6972886966778642, 1.697164416551527]
10	0.01	200	261713	2,00311658849988	[1.6970598076824337, 1.697405694275592]
10	0.001	10	91212	2,00311650818050	[1.6972762900384726, 1.6976720824100715]
10	0.01	100	149592	2,00311650396697	[1.6969817710435702, 1.6973424737044958]
100	0.1	200	303611	2,00311645972200	[1.6971414272097085, 1.6976582621486926]
200	0.1	200	623502	2,00311611317025	[1.6969873035163636, 1.69687930013324]
200	0.01	10	343786	2,00311596201667	[1.6975723631312143, 1.696752740383442]
100	0.1	100	146046	2,00311499417140	[1.6968222592126383, 1.6965360427359366]
200	0.1	100	302494	2,00311476812567	[1.697453845241673, 1.6983224230988605]
10	0.01	10	15994	2,00310690679248	[1.6964035228565246, 1.6993464031728571]
10	0.1	100	15363	2,00309746845663	[1.6961633259164899, 1.7002095158049808]
10	0.1	200	26543	2,00309125665290	[1.7002308031848987, 1.6952521264902063]
200	0.1	10	41484	2,00307423154990	[1.701915498788226, 1.6980037921364244]
100	0.1	10	21386	2,00307295022379	[1.7018389713753017, 1.6960525081820848]
10	0.1	10	1727	2,00068512699929	[1.716552899234296, 1.7267495676862226]

Problem RevAckley

Iterations	Stepsize	Neighbours	Calls to eval	Evaluated value	Best point found
100	0.001	200	7577600	-0,00003440397255	[-9.615795108470324E-6, 7.446930512685743E-6]
200	0.001	100	7613480	-0,00004156967268	[1.0459313042270131E-5, -1.0322195821656701E-5]
200	0.001	200	14644458	-0,00012005030420	[2.3346445116080065E-5, -3.542618577327293E-5]
200	0.01	200	1584483	-0,00012577257391	[4.430972923840276E-5, -3.512346868975874E-6]
10	0.001	200	806423	-0,00013136261137	[4.5140690292823266E-5, -1.0837472097404373E-5]
100	0.001	100	3821133	-0,00015800767958	[-3.628777302399276E-5, -4.243492082111927E-5]
100	0.001	10	431521	-0,00042786490414	[-4.3669878183895644E-5, -1.446082289888928E-4]
200	0.01	100	797495	-0,00061693509258	[1.774994510163129E-4, -1.259986550817974E-4]
200	0.001	10	819459	-0,00075345723450	[-2.3151533452405614E-4, -1.3041915849899236E-4]
200	0.01	10	111796	-0,00160004630035	[-2.909948291428977E-4, -4.816394183073348E-4]
100	0.01	200	816563	-0,00182410864537	[-4.0163289139778097E-4, 4.996371863750281E-4]
100	0.1	100	60600	-0,00484131394360	[-0.0011658226280599204, 0.0012164970780127494]
100	0.01	10	52065	-0,00528029125074	[-0.0017432410106457798, 5.735157528394333E-4]
200	0.1	200	232758	-0,00791337445743	[0.0016429311614975795, -0.002177479429245438]
100	0.1	200	116882	-0,01373517010203	[-0.00308571319474461, 0.0034818001031625325]
200	0.1	100	121704	-0,02662131546958	[0.006373429096177934, 0.005921721205978028]
100	0.1	10	6998	-0,04830812980720	[0.012154704986282419, -0.008742520295722749]
200	0.1	10	15535	-0,06460172571005	[-0.018576348030279785, 0.005351957155834568]
100	0.01	100	432684	-2,57992871764529	[-1.5259236033294545E-4, -0.9520185602536327]
10	0.01	200	71567	-2,57992900099141	[-0.9521604917321626, -2.289444035689804E-4]
10	0.1	200	13880	-2,58037182353494	[0.9502523342852738, 0.00360697005724564]
10	0.01	100	45663	-3,57446304738340	[-0.9689122183816415, 0.9689918466699324]
10	0.1	10	726	-3,58402330596736	[-0.9565679034562102, -0.9832185820251499]
10	0.001	10	42262	-5,38186495890221	[1.9647747884275812, -0.9823456678952447]
10	0.1	100	6576	-5,38257414016655	[-1.9605560526182066, -0.9856862150067746]
10	0.01	10	4147	-6,63778551796862	[0.6005405606959177, 1.8276206915329054]
10	0.001	100	369166	-7,18095170812748	[0.9891507741509146, 2.9672107625023516]