Optimization Report

In Litterman's carbon pricing model, the original optimization consists of 2 phases: genetic algorithm (GA) and gradient search (GS). In order to improve the optimization approach, we introduce alternative methods and integrate them with the original pricing model. Results of all appropriate combinations of these methods, including GA and GS, are given in this documentation, to be compared with the GA+GS benchmark.

Here are some explanations for the variables:

- **Final Utility**:

  Final objective function value of our optimization. This is essentially $-U(0)$ or negative optimized utility at the start point. Our original question has to do with the utility maximization, so we need to minimize $-U(0)$ in our optimization part. In another word, the smaller the final utility $-U(0)$ is, the better.

- **Percentage Decrease**:

  The percentage decrease in average utility of each combination compared to the benchmark, -9.4936. Here, the benchmark is the weighted average of the utility results from RBF+QN, GA+QN (excluding unsatisfactory results), and QN.

- **Error due to Initial Point**:

  This error comes up when we use Quasi-Newton method. In our tests, we take the final utility as the average of all the utility results. These results differ a little bit from each other and we take their standard deviation to be the error here. Notice that no randomness exists in our algorithm except the initial point, so the difference in final utility values must be resulted from that in the initial points.This is what we call error due to initial point here.

- **Stop Constraint**

  3 appropriate stop constraints are listed in most of the tables. Only the one marked in red is the stop constraint that we take in the method in concern.

1 Comparison of Phase 1: GA, RBF, Random

Table 1: Phase 1 Comparison

GA			
Utility after Phase 1	Standard Deviation of Utility	Number of Tests	Average Time
-8.5998	0.9411	100	222.136
RBF			
Utility after Phase 1	Standard Deviation of Utility	Number of Tests	Average Time
-7.9547	0.0337	100	35.85283
Random			
Utility after Phase 1	Standard Deviation of Utility	Number of Tests	Average Time
-8.0585	0.1050	100	3.19

If we compare the 3 phase 1 methods, RBF gives the worst results. GA does a relatively better job, but it has the largest standard deviation and takes much longer time. This is due to the heuristic nature of GA. Indeed, as we can see in the analysis to follow, the advantage of GA is negligible. We can easily make it up with Quasi-Newton and achieve the optimization efficiently.

2 Comparison of Phase 2

Here we compare the phase 2 methods. We do 75 GA iterations and take it as phase 1. The 3 combinations we consider are GA+Fmincon, GA+GS, and GA+Quasi-Newton, whose summary tables are given below. If we look into the final utilities and time summaries, we can see that Quasi-Newton gives us the most favorable outcome. It outperforms not only in the utility result but also in time consumption. Most importantly, Quasi-Newton is the only one that manages to attain the local minimum/maximum. However, Quasi-Newton can give unsatisfactory outcome by accident, please read subsection 2.3 for details.

2.1 GA+Fmincon

When it comes to Fmincon, we use its default settings, including the stop constraint. Upon our observation, Fmincon always stops at stepsize 1×10^{-10}. The 9th test hits the maximum number of iteration (3000), so it exits automatically (exitflag = 0). We therefore take the results of the other 8 tests.

Table 2: GA+Fmincon

Phase 1: GA		
Utility after Phase 1	Consequent Norm of Gradient	Number of Iterations
-8.1343	0.4066	75
Phase 2: Fmincon		
Final Utility	Number of Iterations	Number of Utility Iterations
-9.3634	12.6250	932
Norm of Gradient		
0.3909		
Percentage Decrease	Number of Tests	Error due to Initial Point
1.37%	8	0.0312

As we can see from Table 2, the final utility given by Fmincon is not close enough to the true value for our purpose. Below is the time decomposition for your reference.

Table 3: Time Decomposition (Average) for GA + Fmincon

	Total Time	Self Time
GA_Fmincon	2529.0524	223.716
Fmincon	2301.7846	0.083
Utility_g	2300.7531	2056.272

2.2 GA+GS

The GA+GS combination is EZ-Climate's original method. We fix GS iterations to 200 times.

Table 4: GA+GS

Phase 1: GA		
Utility after Phase 1	Consequent Norm of Gradient	Number of Iterations
-8.4932	0.4415	75
Phase 2: GS		
Final Utility	Final Norm of Gradient	Number of Iterations
-9.4647	0.1640	200
Number of Utility Iterations	Number of Gradient Evaluations	Average Time
200	200	590.5188
Percentage Decrease	Number of Tests	Error of 200 Iterations
0.3%	20	0.0549

Compared to the Fmincon results, GS is much closer to the -9.4936 benchmark, to the first decimal place. We are likely to achieve ideal results with more iterations. Notice here that GA+GS combination gives the greatest standard error though. GS has its advantage with fast speed, but it takes around 10 minutes to complete in Matlab (average time $\approx$ 591 s).

2.3 GA+Quasi-Newton

As we said, Quasi-Newton seems to be the most desirable method among the 3 for phase 2. However, one important phenomenon comes to our attention. Out of the 20 GA+Quasi Newton tests, 17 give results around -9.49 and 3 give those around -8.51. Their average are -9.4936 and 8.5129, respectively. In order to have a better understanding of these 2 conditions, we also summarize them separately in the table to follow.

Table 5: GA+Quasi-Newton

Phase 1: GA		
Utility after Phase 1	Consequent Norm of Gradient	Number of Iterations
-8.4932	0.4415	75
Stop Constraint		
Norm of Gradient$< 10^{-3}$	Iteration$> 500 \times 10^{20}$	fcount $< 10^3$
Phase 2: Quasi-Newton 20		
Final Utility	Final Norm of Gradient	Number of Iterations
-9.3465	8.8202×10^{-4}	221.550
Number of Utility Iterations	Number of Gradient Evaluations	Average Time
536.800	536.800	0.1349
Percentage Decrease	Number of Tests	Error due to Initial Point
1.55%	20	0.3593
[1]: Quasi-Newton 17		
Utility after Phase 1	Final Norm of Gradient	Number of Iterations
-9.4936	8.8510×10^{-4}	218.1765
Number of Utility Iterations	Number of Gradient Evaluations	
530.6471	530.6471	
Percentage Decrease	Number of Tests	Error due to Initial Point
0	17	8.4707×10^{-4}
[2]: Quasi-Newton 3		
Final Utility	Final Norm of Gradient	Number of Iterations
-8.5129	8.6456×10^{-4}	240.6667
Number of Utility Iterations	Number of Gradient Evaluations	
571.6667	571.6667	
Percentage Decrease	Number of Tests	Error due to Initial Point
10.33%	3	1.6189×10^{-4}

If we focus on the 17 results, they give the best final utility, which agrees with the benchmark to the fourth decimal place. Final gradient norm is 8.8510×10^{-4} and finite differentiation error is 8.4707×10^{-4}. Both of them support Quasi-Newton to be a good method. In addition, they on average take 218 iterations, compared to the 932 ones of Fmincon.

When it comes to the 3 results left, you can refer to Section 4 in the back. We compare

these 2 kinds of results by looking into the mitigation levels for all 63 nodes. Approaches to handing the problem are proposed for further studies.

3 Further Analysis with Quasi-Newton

Based on the 2 sections above, we find Quasi-Newton a good candidate for phase 2. In this section, we integrate it with the other 2 phase 1 techniques and compare their outcomes.

3.1 Random + Quasi-Newton

Table 6: Quasi-Newton

Final Utility	Final Norm of Gradient	Number of Iterations
-9.4932	8.6961×10^{-4}	215.8000
Number of Utility Iterations	Number of Gradient Evaluations	
567.7000	567.7000	
Percentage Decrease	Number of Tests	Error due to Initial Point
0.0042%	10	0.0010

Table 7: Time Decomposition (Average) for Quasi-Newton

	Total Time	Self Time
Integrate	1428.9338	24.981
Multiprocessing	1403.744	1403.744
Quasi-Newton	1401.4898	0.0767
Line Search	865.5479	0.0258

Compared with the 17 results given by GA+Quasi-Newton, final utility here differs for 0.0004, which is acceptable. The finite differentiation error here is a little bit greater than that of the GA+Quasi-Newton combination.

3.2 RBF+Quasi-Newton

The RBF+Quasi-Newton is the only combination that gives better utility than the benchmark. Furthermore, the finite differentiation error is even smaller than that of the GA+QN combination. Notice that we test this combination using the GRI computer. This is the main reason it turns out to consume extremely long time, compared to the other methods. See Table 9.

Table 8: RBF + QN

RBF		
Utility after Phase 1	Consequent Norm of Gradient	Number of Iterations
-7.9427	0.3378	20
Stop Constraint		
Norm of Gradient$< 10^{-3}$	Iteration$> 500 \times 10^{20}$	fcount $< 10^3$
Quasi-Newton		
Final Utility	Final Norm of Gradient	Number of Iterations
-9.4939	8.1948×10^{-4}	226
Number of Utility Iterations	Number of Gradient Evaluations	
593.6667	593.6667	
Percentage Decrease	Number of Tests	Error due to Initial Point
-0.00316%	6	7.8043×10^{-4}

Table 9: Time Decomposition (Average) for RBF + Quasi-Newton

	Total Time	Self Time
RBF_QN	7474.1443	20.755
Multiprocessing	7447.9595	7421.8635
Quasi-Newton	7421.8635	0.122
Line Search	4581.0847	0.0393

4 Optimized Mitigation

Below is an analysis of the optimized mitigation levels under GA+Quasi-Newton. In section 2.3, we see final results of GA+Quasi-Newton are divided to 2 kinds, with final utility values around -9.49 and -8.51 separately. Figure 1 summaries the information that we care the most for each individual node:

- **Mean**: the mean mitigation level of all 20 tests.
- **std**: the standard deviation of all 20 tests.
- **17-std**: the standard deviation of the 17 tests, those with $U(0)$ around -9.49.
- **3-std**: the standard deviation of the 3 tests, those with $U(0)$ around -8.51.
- **17ave-3ave**: the difference of the mitigation levels means given by the 2 kind of results. We mark those with absolute value greater than 0.005 in red.

Within results of the first kind (-9.49), we can tell from the small standard deviation (std) that our results have mitigation levels pretty close to each other. In addition, if we look at those 17ave-3ave in red, mitigation levels of these two results agree to a large degree. These 2 kinds of results warn us that the local optimizer can bring about a wrong answer given a bad initiate point. In order to handle this problem, we may constrain the initiate point

in the directions that two results diverge the most. However, we need to do further tests without fixed damage simulations.

Figure 1: final optimized mitigation level analysis

QN				
mean	std	17-std	3-std	17ave-3ave
0.83188	0.00132	0.00132	0.00121	-0.00045
0.98174	0.00265	0.00258	0.00244	-0.00188
0.84524	0.00238	0.00255	0.00028	0.00099
1.27388	0.01270	0.01277	0.00965	-0.00818
1.02554	0.00107	0.00101	0.00125	0.00061
1.04108	0.00306	0.00327	0.00023	0.00140
0.90945	0.00618	0.00666	0.00073	0.00186
1.12032	0.00727	0.00719	0.00627	0.00486
1.08766	0.00704	0.00649	0.00805	0.00566
1.17955	0.00062	0.00064	0.00035	0.00035
1.13842	0.00252	0.00247	0.00255	0.00131
1.20585	0.00181	0.00194	0.00028	0.00076
1.16327	0.00210	0.00227	0.00028	-0.00029
1.20846	0.00395	0.00403	0.00322	0.00135
0.92373	0.01446	0.01566	0.00101	0.00192
1.02268	0.00136	0.00125	0.00163	-0.00102
0.98676	0.00115	0.00105	0.00133	-0.00095
1.02314	0.00212	0.00174	0.00323	-0.00170
0.98504	0.00220	0.00201	0.00281	-0.00136
1.03469	0.00013	0.00013	0.00010	0.00000
0.99808	0.00085	0.00086	0.00006	0.00083
1.04743	0.00319	0.00322	0.00283	-0.00116
1.01101	0.00416	0.00416	0.00375	-0.00199
1.03843	0.00325	0.00348	0.00008	-0.00157
1.00269	0.00377	0.00403	0.00043	-0.00178
1.05316	0.00194	0.00208	0.00010	-0.00067
1.01724	0.00198	0.00209	0.00072	-0.00102
1.06185	0.00438	0.00429	0.00410	-0.00273
1.02662	0.00499	0.00493	0.00467	-0.00274
1.31274	0.01527	0.01647	0.00342	-0.00272
1.20416	0.02117	0.02292	0.00333	-0.00165
0.94198	0.00021	0.00022	0.00014	-0.00005
0.92484	0.00026	0.00017	0.00045	0.00032
0.96128	0.00047	0.00044	0.00059	0.00008
0.93613	0.00046	0.00049	0.00007	-0.00027
0.95946	0.00045	0.00046	0.00040	-0.00014
0.93424	0.00057	0.00058	0.00010	-0.00047
0.97479	0.00120	0.00120	0.00122	-0.00008
0.95591	0.00296	0.00290	0.00279	-0.00190
0.95595	0.00032	0.00032	0.00028	-0.00020
0.93110	0.00053	0.00043	0.00076	-0.00053
0.96977	0.00120	0.00107	0.00060	-0.00181
0.95145	0.00233	0.00237	0.00040	-0.00223
0.96433	0.00087	0.00093	0.00028	0.00022
0.94541	0.00074	0.00078	0.00041	-0.00027
0.98394	0.00204	0.00209	0.00141	0.00109
0.82758	0.11102	0.10657	0.12559	0.04892
0.95507	0.00026	0.00026	0.00023	-0.00008
0.93004	0.00080	0.00084	0.00028	-0.00042
0.96775	0.00109	0.00103	0.00138	-0.00026
0.95012	0.00115	0.00122	0.00067	-0.00024
0.96315	0.00076	0.00069	0.00092	0.00053
0.94320	0.00049	0.00051	0.00027	-0.00009
0.98102	0.00075	0.00068	0.00095	0.00051
0.86250	0.05258	0.03823	0.08142	0.06439
0.96158	0.00078	0.00082	0.00045	-0.00010
0.94053	0.00095	0.00095	0.00078	0.00060
0.97798	0.00152	0.00156	0.00126	0.00036
0.88400	0.03708	0.03996	0.00283	-0.01151
0.99441	0.00633	0.00673	0.00310	0.00104
0.96943	0.00834	0.00901	0.00173	-0.00127
1.08326	0.00767	0.00822	0.00069	-0.00306
0.51492	0.31696	0.25128	0.00000	0.60578