\documentclass[12pt]{article}

\usepackage[utf8]{inputenc}

\usepackage{latexsym}

\usepackage{float}

\usepackage{parskip}

\usepackage{amsfonts}

\usepackage{caption}

\usepackage{commath}

\usepackage{amssymb,amsmath}

\usepackage{graphicx}

\usepackage{multirow}

\usepackage[backend=bibtex,style=numeric,sorting=none]{biblatex}

\usepackage[export]{adjustbox}

\usepackage{subcaption}

\usepackage[top=1in, bottom=1in,left=1in, right=1in]{geometry}

\usepackage{minted}

\usepackage{color}

\newcommand{\tb}{\textcolor{blue}}

\newcommand{\tc}{\textcolor{red}}

\newcommand{\bs}{\boldsymbol}

\newenvironment{alphafootnotes}

{\par\edef\savedfootnotenumber{\number\value{footnote}}

\renewcommand{\thefootnote}{\alph{footnote}}

\setcounter{footnote}{0}}

{\par\setcounter{footnote}{\savedfootnotenumber}}

\begin{document}

\begin{center}

\Large\bf{Optimization Report}

\end{center}

\normalsize

\pagenumbering{arabic}

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:

\begin{itemize}

\item \textbf{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.

\item \textbf{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.

\item \textbf{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.

\item \textbf{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.

\end{itemize}

\section{Comparison of Phase 1: GA, RBF, Random}

\begin{center}

\captionof{table}{Phase 1 Comparison}

\begin{tabular}{||c c c c||}

\hline

& \textbf{GA} & &\\ [0.5ex]

\hline\hline

Utility after Phase 1 & Standard Deviation of Utility & Number of Tests & Average Time\\

\hline

-8.5998 & 0.9411 & 100 & 222.136\\

\hline\hline

& \textbf{RBF} & &\\ [0.5ex]

\hline\hline

Utility after Phase 1 & Standard Deviation of Utility & Number of Tests & Average Time\\

\hline

-7.9547 & 0.0337 & 100 & 35.85283\\

\hline\hline

& \textbf{Random} & &\\ [0.5ex]

\hline\hline

Utility after Phase 1 & Standard Deviation of Utility & Number of Tests & Average Time \\

\hline

-8.0585 & 0.1050 & 100 & 3.19\\

\hline

\end{tabular}

\label{comparison}

\end{center}

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.

\section{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.

\subsection{GA+Fmincon}

When it comes to Fmincon, we use its default settings, including the stop constraint. Upon our observation, Fmincon always stops at stepsize $1\times 10^{-10}$. 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.

\begin{center}

\captionof{table}{GA+Fmincon}

\begin{tabular}{||c c c||}

\hline

& \textbf{Phase 1: GA} & \\ [0.5ex]

\hline\hline

Utility after Phase 1 & Consequent Norm of Gradient & Number of Iterations\\

\hline

-8.1343 & 0.4066 & 75\\

\hline\hline

& \textbf{Phase 2: Fmincon} &\\ [0.5ex]

\hline\hline

Final Utility & Number of Iterations & Number of Utility Iterations\\

\hline

-9.3634 & 12.6250 & 932\\

\hline

Norm of Gradient & &\\

\hline

0.3909 & & \\

\hline\hline

Percentage Decrease & Number of Tests & Error due to Initial Point\\

\hline

1.37\% & 8 & 0.0312\\

\hline

\end{tabular}

\label{gafmin}

\end{center}

As we can see from Table \ref{gafmin}, 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.

\begin{center}

\captionof{table}{Time Decomposition (Average) for GA + Fmincon}

\begin{tabular}{||c c ||}

\hline

& Self Time \\ [0.5ex]

\hline\hline

GA & 223.716 \\

\hline

Fmincon & 0.083\\

\hline

Utility\\_g & 2056.272\\[0.5ex]

\hline

\end{tabular}

\label{time\_fmincon}

\end{center}

\subsection{GA+GS}\label{gags}

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

\begin{center}

\captionof{table}{GA+GS}

\begin{tabular}{||c c c ||}

\hline

& \textbf{Phase 1: GA} & \\ [0.5ex]

\hline\hline

Utility after Phase 1 & Consequent Norm of Gradient & Number of Iterations \\

\hline

-8.4932 & 0.4415 & 75\\

\hline\hline

& \textbf{Phase 2: GS} &\\ [0.5ex]

\hline\hline

Final Utility & Final Norm of Gradient & Number of Iterations\\

\hline

-9.4647 & 0.1640 & 200 \\

\hline

Number of Utility Iterations & Number of Gradient Evaluations & Average Time\\

\hline

200 & 200 & 590.5188\\

\hline\hline

Percentage Decrease & Number of Tests & Error of 200 Iterations\\

\hline

0.3\% & 20 & 0.0549\\

\hline

\end{tabular}

\label{gags}

\end{center}

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).

\subsection{GA+Quasi-Newton}

\label{sec:gaqn}

As we said, Quasi-Newton seems to be the most desirable method among the 3 for phase 2. Out of the 20 GA+Quasi Newton tests, we get 17 qualified ones. The other 3 fail because the penalty for negative mitigation levels influences the value of our objective function to a large degree. The average of the 17 samples is -9.4936. We summarize them in the table to follow.

\begin{center}

\captionof{table}{GA+Quasi-Newton}

\begin{tabular}{||c c c ||}

\hline

& \textbf{Phase 1: GA} & \\ [0.5ex]

\hline\hline

Utility after Phase 1 & Consequent Norm of Gradient & Number of Iterations \\

\hline

-8.4932 & 0.4415 & 75\\

\hline\hline

& \textbf{Stop Constraint} & \\ [0.5ex]

\hline\hline

\tc{Norm of Gradient$ < 10^{-3}$} & Iteration$ > 500\times 10^{20}$ & fcount $< 10^3$\\

\hline\hline

& \textbf{Phase 2: Quasi-Newton 17} &\\ [0.5ex]

\hline\hline

Utility after Phase 1 & Final Norm of Gradient & Number of Iterations\\

\hline

-9.4936 & $8.8510 \times 10^{-4}$ & 218.1765\\

\hline

Number of Utility Iterations& Number of Gradient Evaluations & Average Time\\

\hline

530.6471 & 530.6471 & 0.1349\\

\hline\hline

Percentage Decrease & Number of Tests & Error due to Initial Point\\

\hline

0 & 17 & $8.4707\times 10^{-4}$\\

\hline

\end{tabular}

\label{gaqn}

\end{center}

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 $8.8510 \times 10^{-4}$ and finite differentiation error is $8.4707\times 10^{-4}$. Both support Quasi-Newton to be a good method. In addition, they on average take 218 iterations, compared to the 932 ones of Fmincon.

\section{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.

\subsection{Random + Quasi-Newton}

\begin{center}

\captionof{table}{Quasi-Newton}

\begin{tabular}{||c c c||}

\hline

Final Utility & Final Norm of Gradient & Number of Iterations \\

\hline

-9.4932 & 8.6961$\times 10^{-4}$ & 215.8000\\

\hline

Number of Utility Iterations & Number of Gradient Evaluations &\\

\hline

567.7000 & 567.7000 & \\

\hline\hline

Percentage Decrease & Number of Tests & Error due to Initial Point\\

\hline

0.0042\% & 10 & 0.0010\\

\hline

\end{tabular}

\label{qn}

\end{center}

\begin{center}

\captionof{table}{Time Decomposition (Average) for Quasi-Newton}

\begin{tabular}{||c c ||}

\hline

& Self Time \\ [0.5ex]

\hline\hline

Damage Simulation & 24.981 \\

\hline

Calling utility and gradient & 1403.744 \\

\hline

Quasi-Newton & 0.0767\\

\hline

Line Search & 0.0258\\[0.5ex]

\hline

\end{tabular}

\label{time\_qn}

\end{center}

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.

\subsection{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 \ref{time\_rbfqn}.

\begin{center}

\captionof{table}{RBF + QN}

\begin{tabular}{||c c c||}

\hline

& \textbf{RBF} & \\ [0.5ex]

\hline\hline

Utility after Phase 1 & Consequent Norm of Gradient & Number of Iterations\\

\hline

-7.9427 & 0.3378 & 20\\

\hline\hline

& \textbf{Stop Constraint} & \\ [0.5ex]

\hline\hline

\tc{Norm of Gradient$ < 10^{-3}$} & Iteration$ > 500\times 10^{20}$ & fcount $< 10^3$\\

\hline\hline

& \textbf{Quasi-Newton} &\\ [0.5ex]

\hline\hline

Final Utility & Final Norm of Gradient & Number of Iterations \\

\hline

-9.4939 & $8.1948\times 10^{-4}$ & 226\\

\hline

Number of Utility Iterations & Number of Gradient Evaluations & \\

\hline

593.6667 & 593.6667 & \\

\hline\hline

Percentage Decrease & Number of Tests & Error due to Initial Point\\

\hline

-0.00316\% & 6 & $7.8043\times 10^{-4}$\\

\hline

\end{tabular}

\label{rbfqn}

\end{center}

\begin{center}

\captionof{table}{Time Decomposition (Average) for RBF + Quasi-Newton}

\begin{tabular}{||c c ||}

\hline

& Self Time \\ [0.5ex]

\hline\hline

RBF & 20.755 \\

\hline

Calling utility and gradient & 7421.8635\\

\hline

Quasi-Newton & 0.122\\

\hline

Line Search & 0.0393\\[0.5ex]

\hline

\end{tabular}

\label{time\_rbfqn}

\end{center}

\section{Sensitivity Analysis with Quasi-Newton}

\begin{center}

\captionof{table}{Quasi-Newton Sensitivity Analysis}

\begin{tabular}{||c c c||}

\hline

Final Utility & Final Norm of Gradient & Number of Iterations \\

\hline

-9.4942 & 8.6500$\times 10^{-4}$ & 214.6897\\

\hline

Number of Utility Iterations & Number of Gradient Evaluations &\\

\hline

549.4333 & 549.4333 & \\

\hline\hline

Standard Deviation of Final Utility & Number of Tests & Total Time Average\\

\hline

0.0011 & 29 & 1472.2\\

\hline

\end{tabular}

\label{qn}

\end{center}

\end{document}