Revision date: 07/26/2021

---
0. Contents

---
1. About the program

The first version of the TikTak Global Optimization algorithm was written by Fatih Guvenen and described in Guvenen (2011). The code has been improved over time, with key contributions by Anthony Smith, Serdar Ozkan, and Fatih Karahan. The code provided here has been modified from earlier versions by Arun Kandanchatha and Serdar Ozkan. A description of the basic version of the TikTak algorithm can be found in Arnoud, Guvenen, and Kleineberg (2019).

This version of the TikTak code contains two important improvements over previous versions:

(i) It contains the most efficient implementation of TikTak to date (including relative to the version benchmarked in Arnoud, Guvenen, and Kleineberg (2019)).
(ii) It can be run in parallel mode out of the box without requiring any specialized software (MPI, OpenMP, etc.). It can be run both on computer clusters and on fully distributed mixed computational environments (e.g., PCs running Windows, Linux, or OSX in different locations) using a syncing solution like DropBox.

Great care was taken to make it as compliant with Fortran 90 as possible, but there may be a few invocations to Fortran 95 intrinsics.

References:

1. Guvenen, Fatih (2011): "Macroeconomics with Heterogeneity: A Practical Guide", FRB Richmond Economic Quarterly, Volume 97, Number 3, pp. 255–326.
2. Guvenen, Fatih and Anthony Smith (2014): “Inferring Labor Income Risk and Partial Insurance from Economic Choices,” Econometrica, November 2014, 82 (6), 2085–2129.
3. Guvenen, Fatih, Serdar Ozkan, and Jae Song, “The Nature of Countercyclical Income Risk,” Journal of
Political Economy, 2014, 122 (3), 621–660.
4. Arnoud, Antoine, Fatih Guvenen, Tatjana Kleineberg (2019): "Benchmarking Global Optimizers", NBER Working Paper, 26340.
5. Guvenen, Fatih, Fatih Karahan, Serdar Ozkan, and Jae Song (2021): “What Do Data on Millions of U.S. Workers Reveal About Lifecycle Earnings Dynamics?,” Econometrica, forthcoming.

```
-----------------------------------
2. Executing the program
-----------------------------------
To execute the program, run
   ./GlobalSearch <-1|0|1|2|4|5> configfile <a|b|d>
For help, run
   ./GlobalSearch

-1 = exit state: end all running instances

0 = cold start - The first invocation of the program, that will set up all the parallel helper
files.
            Should always be the parameter when this is the first attempt at solving the
problem.
1 = warm start - after one process is already running, all helper programs should be invoked
using
            a warm start.
2 = update number of Sobol points - update the sobol point parameters over which to search
             as well as the local optimizations from these sobol points, but assume everything
             else in the config file has not been changed.
3 = update number of local minimizations - Increase the number of local minimizations but keep
            everything else same in the config file. This option uses the results from
previously
            found local minimums.
4 = Just evaluate the objective function once for given initial guess in the config file with
            diagnostic option.
5 = Run local minimization once for given initial guess in the config file.

a = Runs AMOEBA (Simplex algorithm) for local minimization
b = Runs BOBYQA (DFNLS algorithm) for local minimization
d = Runs DFPMIN (BFGS Quasi-Newton method) for local minimization

-----------------------------------
3. Description of source files
-----------------------------------
These files are specific for the generic search; you are not expected to make any changes.
GlobalSearch.f90  - the main driver program for the search.
genericParams.f90 - the parameters that the generic search program needs. Note that
             we do not put function specific parameters in this file
minimize.f90      - this module contains the code for minimization.
nrtype.f90        - basic types used in all functions.
simplex.f90       - open source code that obtains an m-dimensional simplex centered
                    on the origin. Used for amoeba search
stateControl.f90  - module that manages the genericSearch states using file I/O.
utilities.f90     - implementation of sobol and other helper functions.
---------------------------------
These are all specific to the value function being solved. This files needs to be specified by
the user.
objective.f90     - the specific objective function being solved. Require the following
functions
                    to be defined: objFun, dfovec, obj_initialize, diagnostic. All model
specific
                          parameters are also defined within this file.

-----------------------------------
4. Description of text files
```

```
-----------------------------------
config.txt        - the configuration file for execution---input file that needs to be
specified by the user.
readme.txt        - this file
logfile.txt       - the log of the program execution---output file

-----------------------------------
5. Description of .dat files
-----------------------------------
internalConfig.dat- a copy of the config.txt file, but for use by parallel instances
seq.dat           - the number of the last concurrent instance started
lastParam.dat     - the last initial point being used for local minimization
lastSobol.dat     - the last sobol point for which objective function being evaluated
legitSobol.dat    - number of sobol points evaluated with objective values less than
                          maximum legitimate objective value (defined in config.txt)
missingSobol.dat  - Vector of numbers of sobol points that are not evaluated after
                          finishing evaluation of sobol points (may be due to some instances
being stopped).
                          These sobol points are re-visited to be evaluated.
sobol.dat         - the list of sobol points
sobolFnVal.dat    - the value of the objective function at each sobol point
state.dat         - the current state of the program
x_starts.dat      - sorted values of sobolFnVal, to be used by local minimization routine to
converge
                    to global minimum
searchResults.dat - the minimized objective function and associated parameters.
                          Each line corresponds to one set of parameters and show the following
info:
                          the instance number running the code, the number of the sobol point,
                          how many global minima found so far, obecjtive value, and associated
parameters.
searchStart.dat   - the starting point in the local minimization for the associated line in
                    searchResults.dat. Each line shows the following information:
                          the instance number running the code, the number of the sobol point,
                          parameter values for the starting point.
FinalResults.dat -  the minimized objective function and associated parameters
                          after running a local minimization around the best global minimum so
far
                          one more using BOBYQA.
                          Each line corresponds to one set of parameters and show the following
info:
                          the instance number running the code, the number of the sobol point,
                          obecjtive value, and associated parameters.
FinalStart.dat   -  the best global minimum so far as the starting point for
                          running a local minimization one more using BOBYQA.
                          Each line corresponds to one set of parameters and show the following
info:
                          the instance number running the code, the number of the sobol point,
                          and associated parameters.

-----------------------------------
6. Specifying the objective function
-----------------------------------
The objective function should be specified in the file "objective.f90". Require the following
functions
to be defined: objFun, dfovec, obj_initialize, diagnostic

a: amoeba
The amoeba routine takes a function of the following form:
```

```
        FUNCTION objFunc(theta)
            use genericParams
            use nrtype
                implicit none
            REAL(DP),DIMENSION(p_nx),INTENT(IN) :: theta
            REAL(DP) :: objFunc
        END FUNCTION objFunc
```

b: bobyq
The bobyq routine requires a function named dfovec. From the comments of the bobyq code,

```
            SUBROUTINE dfovec(n, mv, x, v_err)
```

It must provide the values of the vector function v_err(x) : R^n to R^{mv}
at the variables X(1),X(2),...,X(N), which are generated automatically in
a way that satisfies the bounds given in XL and XU.

d: DFPMIN
The DFPMIN procedure minimizes a user-written function Func of two or more independent variables using the Broyden-Fletcher-Goldfarb-Shanno variant of the Davidon-Fletcher-Powell method, using its gradient.

```
    FUNCTION func(x)
      USE UTILITIES, ONLY: DP
      IMPLICIT NONE
      REAL(DP), DIMENSION(:), INTENT(IN) :: x
      REAL(DP) :: func
    END FUNCTION func
```