The AxParafit and AxPcoords Manual

A. Stamatakis[1], A. Auch[2], J. Meier-Kolthoff[2], and M. Göker[3]

[1] École Polytechnique Fédérale de Lausanne
School of Computer & Communication Sciences
Laboratory for Computational Biology and Bioinformatics (LCBB)
Alexandros.Stamatakis@epfl.ch
[2] Center for Bioinformatics (ZBIT),
University of Tübingen, Germany
[3] Organismic Botany/Mycology,
University of Tübingen, Germany

1 About AxParafit and AxPcoords

AxPcoords and AxParafit are highly optimized versions of Pierre Legendre's Parafit [1] and DistPCoA [2] programs that implement a statistical test for host-parasite co–evolution.

AxPcoords is 8–26 times faster than DistPCoA and numerically stable on large datasets. AxParafit runs 5–61 times faster than Parafit with a lower memory footprint (up to 35%) while the performance benefit increases with growing dataset size.

AxParafit has also been parallelized with MPI (Message Passing Interface, www-unix.mcs.anl.gov/mpi/index.htm) since it is much more compute-intensive than AxPcoords. The MPI-based parallel implementation of AxParafit shows good scalability on up to 128 processors, even on medium-sized datasets.

The large performance improvement is achieved via porting the original program to C, manual code optimization, and integration of highly efficient BLAS (Basic Linear Algebra Package, www.netlib.org/blas/) routines into AxParafit and highly efficient LAPACK (Linear Algebra PACKage, www.netlib.org/lapack/) routines into AxPcoords respectively.

AxParafit and AxPcoords have also been integrated into the CopyCat tool ([3], www-ab.informatik.uni-tuebingen.de/software/copycat/welcome.html), that provides an easy-to-use Graphical User Interface for AxParafit-based analyses.

2 Installation, Compilers, Platforms

2.1 Obtaining the Code

AxParafit/AxPcoords are available at icwww.epfl.ch/~stamatak/AxParafit.html. This page also offers pre-compiled binaries for several platforms.

2.2 Requirements

As already mentioned the programs make use of highly tuned linear algebra packages for optimal performance.

AxParafit can be compiled without those libraries, but the expected performance degradation is around factor 4–5. To achieve optimal performance for AxParafit you will need to download and install one of the following Math Libraries: Intel Math Kernel Library (Intel MKL: www.intel.com/cd/software/products/asmo-na/eng/307757.htm, free for academic use), AMD Core Math Library (AMD ACML: developer.amd.com/acml.jsp, free for academic use) or the Automatically Tuned Linear Algebra Software (ATLAS: math-atlas.sourceforge.net, free non-commercial software). To compile the parallel version of AxParafit an MPI `mpicc`-compiler is required. We recommend the MPICH package: www.mcs.anl.gov/mpi/mpich.

AxPcoords needs an external library in any case to compute eigenvectors and eigenvalues. It can either use AMD ACML or Intel MKL (see above) or the respective functions implemented in the GNU Scientific Library (GNU GSL: www.gnu.org/software/gsl, free non-commercial software). Note that, performance degradations are expected when using GNU GSL.

2.3 Compiling & Installing

To install AxParafit/AxPcoords download the `AxParafit.dist.tar.gz` and `AxPcoords.dist.tar.gz` archives and uncompress them.

If you want to use AxParafit without highly optimized libraries type `make AxParafit` to compile the sequential and `make AxParParafit` to compile the parallel *unoptimized* versions of the program. You will obtain two executables `AxParafit` and `AxParParafit` which you can then put e.g. into your Linux PATH.

If you want to use AxParafit with optimized libraries install Intel MKL, AMD ACML, or ATLAS first. The next step will be to adapt some variables in the AxParafit `Makefile`. The variables you have to adapt are called `BLAS_CFLAGS` and `BLAS`. Note that there is a different version of those variables for MKL, ACML, and ATLAS. Make sure, you comment out those `BLAS_CFLAGS` and `BLAS` variables by using `#` for the libraries you don't want to use. Once you have done that you can then go ahead and modify the `-I` path in `BLAS_CFLAGS` to point to your local installation and also the `-L` path in `BLAS` to point to your local installation.

Once this is done you can type `make AxParafitBLAS` to compile the sequential version and `make AxParParafitBLAS` to compile the parallel version. You will obtain two executables `AxParafitBLAS` and `AxParParafitBLAS` which you can then put e.g. into your Linux PATH. Note, that for running the parallel version the `AxParParafit` executable needs to be preceeded by the respective MPI command (mostly `mpiexec` or `mpirun`) with appropriate parameters such as the number of CPUs you want to use. Since this is very

cluster- and installation-specific, please ask the administrator of your cluster about it.

The procedure for AxPcoords is very similar. You will first have to install GNU GSL, Intel MKL, or AMD ACML. Then—depending on which library you have installed—you should adapt the `-I` and `-L` paths in the respective variables `GSL/GSL_CFLAGS`, `MKL/MKL_CFLAGS`, `ACML/ACML_CFLAGS`. To compile, once again depending on the library you use type `make AxPcoordsGSL`, `make AxPcoordsMKL`, or `make AxPcoordsACML` and you will get binaries named `AxPcoordsGSL`,`make AxPcoordsMKL`, `make AxPcoordsACML`.

3 The AxParafit Options

Note, that there is a different set of options for the sequential and parallel programs because the parallel program only executes the much more compute-intensive test on individual host–parasite links. Thus, the parallel version *must* read–in a file (called `tracefile`) with data from the global test of co–speciation that has to be computed using the sequential version.

Note, that options `n1, n2, n3, n4, A, B, C` correspond one-to-one to the inputs that have to be made in the standard Parafit program.

WARNING: The -p option does not correspond to the original Parafit option. in AxParafit a setting of -p 100 corresponds to executing 99 permutations in Parafit!

For details and explanations with respect to the meaning of these parameters please refer to the original Parafit manual at www.bio.umontreal.ca/Casgrain/en/labo/parafit.html.

```
AxParafit[BLAS] -p numberOfPermutations -n1 N1 -n2 N2 -n3 N3
                -n4 N4 -A associationMatrix -B parasiteMatrix
                -C hostMatrix -n runID
                [-g] [-h] [-t traceFileName]

        -p     Specify the number of permutations you want
               to execute.

        -n1    Specify number of Rows in associationMatrix
        -n2    Specify number of Columns in associationMatrix
        -n3    Specify number of Rows in hostMatrix
        -n4    Specify number of Columns in parasiteMatrix

        -A     Specify file name of association matrix
        -B     Specify file name of parasite matrix
        -C     Specify file name of host matrix

        -n     Specify a run Name/ID for this run which will be
               appended to all output files

        -g     Execute global test of cospeciation only and exit.
               This will produce an appropriate input file for the
               parallel program called tracefile.runID

               DEFAULT: OFF

        -h     Display this help message

        -t     Specify the name of a binary trace-file written by
               a global cospeciation test with AxParafit.
               The program will start executing individual
               tests of cospeciation directly.
```

AxParafit will write two output files, a *binary* file called `tracefile.runID` and an output file which corresponds to the Parafit output file which is called `outfile.runID`.

```
AxParParafit[BLAS] -p numberOfPermutations -n1 N1 -n2 N2 -n3 N3
                   -n4 N4 -A associationMatrix -B parasiteMatrix
                   -C hostMatrix
                   -n runID -t traceFileName [-h]

        -p     Specify the number of permutations you want to execute.

        -n1    Specify number of Rows in associationMatrix
        -n2    Specify number of Columns in associationMatrix
        -n3    Specify number of Rows in hostMatrix
        -n4    Specify number of Columns in parasiteMatrix

        -A     Specify file name of association matrix
        -B     Specify file name of parasite matrix
        -C     Specify file name of host matrix

        -n     Specify a run Name/ID for this run which will be
               appended to all output files

        -t     Specify the name of a binary trace-file written by a
               global cospeciation test with sequential AxParafit[BLAS].

        -h     Display this help message
```

AxParParafit will write only one output file called `outfile.runID` which corresponds to the original Parafit output file.

4 The AxPcoords Options

The input matrix and the `-n` parameter are the same as in the original DistPCoA implementation. The option `-t` to automatically transpose the output matrix has mainly been implemented since the test in Parafit and AxParafit requires the host matrix C to be transposed. This also facilitates the integration into CopyCat.

```
AxPcoords[GSL|ACML|MKL] -f inputMatrixName -n rowNumber [-h]
                        [-t]

        -f     Specify file Name of quadratic input matrix

        -n     Specify number of Rows in quadratic input
               Matrix

        -h     Display this help message

        -t     Specify if the output matrix shall be
               transposed

               DEFAULT: OFF
```

WARNING: DistPCoA provides two additional methods (Lingoes and Cailliez) to correct for negative eigenvalues. In contrast to this, AxPcoords simply discards eigenvectors with negative eigenvalues. The rationale for this is that negative eigenvalues typically tend to be very small. Thus, the substantial additional computational effort to correct very small negative eigenvalues via the Lingoes or Cailliez methods can be avoided, due to their marginal influence, particularly on large datasets.

5 An Example Analysis with AxParafit

To test and get used to AxParafit we provide an archive with test data at icwww.epfl.ch/~stamatak/AxParafit.html. You can download it and uncompress it to test AxParafit. The archive contains three data files called `smallA`, `smallB`, and `smallC`.

To execute an AxParafit run with 9 permutations you can type:

```
AxParafitBLAS -p 10 -n1 139 -n2 430 -n3 421 -n4 136
              -A smallA -B smallB -C smallC -n TEST
```

Once again note that `-p 10` corresponds to 9 permutations!

Here, we execute `AxParafitBLAS` that uses one of the highly optimized linear algebra packages.

The output should look something like this:

```
 Global test of cospeciation:                    ParaFitGlobal = 141710945.81848   Prob  = 0.10000

Parasite  1  Host 336   F1 =   716557.30539   Prob1 =  0.10000   F2 =    0.00411   Prob2 =  0.10000
Parasite  1  Host 337   F1 =   716760.94339   Prob1 =  0.10000   F2 =    0.00411   Prob2 =  0.10000
Parasite  1  Host 338   F1 =   649660.46703   Prob1 =  0.10000   F2 =    0.00372   Prob2 =  0.10000
Parasite  1  Host 349   F1 =   650444.72648   Prob1 =  0.10000   F2 =    0.00373   Prob2 =  0.10000
Parasite  1  Host 350   F1 =   650444.73681   Prob1 =  0.10000   F2 =    0.00373   Prob2 =  0.10000
......
```

The output files of this run will be called `outfile.TEST` and `tracefile.TEST`.

Now, if you want to use the parallel version to conduct the individual test of host-parasite links you would proceed as follows: First just execute the global test with the sequential version by adding `-g` to the above command. Don't forget to give this run a new name with e.g. `-n GLOBAL`.

```
AxParafitBLAS -g -p 10 -n1 139 -n2 430 -n3 421 -n4 136
              -A smallA -B smallB -C smallC -n GLOBAL
```

The output should look something like this:

```
 Global test of cospeciation:                    ParaFitGlobal = 141710945.81848   Prob  = 0.10000

Global Significance Computed, exiting ....
```

Then you can start the parallel computation by passing `tracefile.GLOBAL` to AxParParafitBLAS via the `-t` switch. Note the installation-specific MPI commands `mpirun_rsh -np 4 -hostfile hostfile` before AxParParafit-BLAS.

WARNING: it might not be a good idea to perform the sequential computation of the tracefile on a different computer architecture than the parallel test of individual links. This file is a binary file and formats can vary across systems.

```
mpirun_rsh -np 4 -hostfile hostfile AxParParafitBLAS -p 10 -n1 139 -n2 430 -n3 421
             -n4 136 -A smallA -B smallB -C smallC -t tracefile.GLOBAL -n PARALLEL
```

The output should look something like this:

```
READING trace file tracefile.GLOBAL
Global test of cospeciation:                    ParaFitGlobal = 141710945.81848   Prob  = 0.10000

Parasite  1  Host 336   F1 =   716557.30539   Prob1 =  0.10000   F2 =    0.00411   Prob2 =  0.10000
Parasite  1  Host 337   F1 =   716760.94339   Prob1 =  0.10000   F2 =    0.00411   Prob2 =  0.10000
Parasite  1  Host 338   F1 =   649660.46703   Prob1 =  0.10000   F2 =    0.00372   Prob2 =  0.10000
Parasite  1  Host 349   F1 =   650444.72648   Prob1 =  0.10000   F2 =    0.00373   Prob2 =  0.10000
Parasite  1  Host 350   F1 =   650444.73681   Prob1 =  0.10000   F2 =    0.00373   Prob2 =  0.10000
Parasite  1  Host 352   F1 =   650444.73539   Prob1 =  0.10000   F2 =    0.00373   Prob2 =  0.10000
Parasite  1  Host 354   F1 =   650444.74820   Prob1 =  0.10000   F2 =    0.00373   Prob2 =  0.10000
........
```

References

1. Legendre, P., Desdevises, Y., Bazin, E.: A statistical test for host-parasite coevolution. Systematic Biology **51** (2002) 217–234
2. Legendre, P., Anderson, M.J.: Program distpcoa. Département de sciences biologiques, Université de Montréal (1998) 10 pages
3. Meier-Kolthoff, J.P., Auch, A.F., Huson, D.H., Göker, M.: Copycat: Cophylogenetic analysis tool. Bioinformatics (2007) Advance on-line access.