Paramotopy

Prepared by
Daniel Brake
Department of Mathematics

Colorado State University
Fort Collins, Colorado

Contents

1 Introduction

The Paramotopy program is a linux/unix wrapper around Bertini which permits rapid parallel solving of parameterized polynomial systems. It uses MPI for inter-process communication.

Paramotopy consists of two executable programs: Paramotopy, and Step2, and further depends on having a copy of the parallel version of Bertini. Paramotopy is called from the command line, and in turn calls Bertini and Step2.

1.1 Contact

For assistance with Paramotopy, please contact Daniel Brake at `danielthebrake@gmail.com`.
For help with Bertini, contact Daniel Brake `danielthebrake@gmail.com`, Dan Bates `bates@math.colostate.edu`, or one of the other Bertini authors.

disclaimer

Paramotopy and all related code, executables, and other material is offered without warranty, for any purpose, implied or explicit.

2 Getting Started

2.1 Compilation and Installation

Paramotopy has the following library dependencies:

- tinyxml (lies internally to the program's folders, as per the license agreement).
- MPI, version2 (for parallelism)
- boost – regex, system, filesystem (for file manipulation)
- openmp (for the most accurate timing available).

In addition, Paramotopy relies on a suitably compiled version of Bertini, which has the following dependencies:

- mpfr
- gmp
- bison
- flex
- MPI

When Bertini is compiled for use with Paramotopy, it must be compiled in two ways: into an executable, and a library. When compiled into the executable, it must have the flag `-D_HAVE_MPI`. In contrast, the library *must not* have the `-D_HAVE_MPI` flag. Compilation is a little different of every machine, of course, and a long term goal is to use something like `./configure` to handle setting everything up properly for start-to-finish with minimal user input.

3 Input Files

The input file for Paramotopy is fairly simple. Examples appear in InputFiles(**??**,**??**) It consists of a top line declaring the numbers of: functions, variable groups, parameters, and constants. Next come the functions, without any name declarations, or terminating semi-colons. Variable groups come next, with each group getting its own line; variables are comma separated, and there is *no terminating character*. Then come constant declaration, with the word `constant` appearing before a comma separated, semi-colon terminated line containing the constant names. Each constant gets its own line, with `name = value;` being the format. Finally, the type of solve is indicated: 1 asserts the user supplies a text file containing parameter value, 0 indicates the computer will supply a mesh. If it is userdefined, the next line is the name of the file, and the final lines simply indicate the parameter names. If it is computer-generated mesh, the final lines tell the name, starting point of discretization, ending point of discretization, and number of discretization points.

Paramotopy has minimal error correction in the portion of the program, so a error in an input file is likely to cause a fairly benign (will not corrupt data) program crash. If the program is able to parse the input file without errors, it will display information to the screen and user may check everything has been imported correctly.

```
7  1  2  21
Fa-k1*a*b+k2*c-k4*a*f+k5*f-Da*a
Fb-k1*a*b+k2*c+k8*g-k9*b*f-Db*b
Fc+k2*a*b-k2*c-k3*c-k6*c*e+k7*g-Dc*c
Fd+k3*c-Dd*d
Fe-k4*a*e+k5*f-k6*c*e+k7*g-De*e
Ff+k4*a*e-k5*f+k8*g-k9*b*f-Df*f
Fg+k6*c*e-k7*g-k8*g+k9*b*f-Dg*g
a,b,c,d,e,f,g
constant k3,k1,k5,k6,k7,k8,k9,Fa,Fb,Fc,Fd,Fe,Ff,Fg,Da,Db,Dc,Dd
    ,De,Df,Dg;
k3=.80349;
k1=.980389;
k5=.8034;
k6=.8018;
k7=.16876;
k8=.7982;
k9=.58973;
Fa=.4264;
Fb=.5284;
Fc=.1687;
Fd=.167896;
Fe=.5673;
Ff=.69386;
Fg=.79827;
Da=.0692;
Db=.08762;
Dc=.2897;
Dd=.0828;
De=.26967;
Df=.4238;
Dg=.5872;
0
k4    0  0  1  0  40
k2    0  0  1  0  50
```

InputFile 1: Paramotopy input file demonstrating use of user-defined parameter file. Line 1 indicates 12 equations (lines 2-13), in 1 variable group (line 14), with five parameters (lines 33-34), and 21 constant declarations (10-31). On line 32, `0` tells Paramotopy to make a mesh from the parameters discretized in lines 33-34. Parameter `k4` will be broken into 40 points on the complex line segment $0+0i$ to $1+0i$. Similarly, `k2` will be broken into 50 points, so a solve using this input file would have 2000 points total in the Step2 run.

```
12 1 5 0
0.4318*c1*c2 - 0.2435*s1 + 0.0934*a*s1 + 0.0203*a*c1*s2*s3 -
    0.0203*a*c1*c2*c3 - (0.4318*c4*c5 - 0.1501*s4 - 0.0203*c4*
    c5*c6 + 0.4331*c4*c5*s6 + 0.4331*c4*c6*s5 + 0.0203*c4*s5*s6
    +delta)
0.2435*c1 - 0.0934*a*c1 + 0.4318*c2*s1 + 0.0203*a*s1*s2*s3 -
    0.0203*a*c2*c3*s1 - (0.1501*c4 + 0.4318*c5*s4 - 0.0203*c5*
    c6*s4 + 0.4331*c5*s4*s6 + 0.4331*c6*s4*s5 + 0.0203*s4*s5*s6
    )
0.0203*a*c2*s3 - 0.4318*s2 + 0.0203*a*c3*s2 - (0.4331*c5*c6 -
    0.4318*s5 + 0.0203*c5*s6 + 0.0203*c6*s5 - 0.4331*s5*s6)
0.4318*c1*c2 - 0.1501*s1 - 0.0203*c1*c2*c3 + 0.4331*c1*c2*s3 +
     0.4331*c1*c3*s2 + 0.0203*c1*s2*s3 - x
0.1501*c1 + 0.4318*c2*s1 - 0.0203*c2*c3*s1 + 0.4331*c2*s1*s3 +
     0.4331*c3*s1*s2 + 0.0203*s1*s2*s3 - y
0.4331*c2*c3 - 0.4318*s2 + 0.0203*c2*s3 + 0.0203*c3*s2 -
    0.4331*s2*s3 - z
s1^2+c1^2-1
s2^2+c2^2-1
s3^2+c3^2-1
s4^2+c4^2-1
s5^2+c5^2-1
s6^2+c6^2-1
s1,c1,s2,c2,s3,c3,s4,c4,s5,c5,s6,c6
1
robomc_10000
a
delta
x
y
z
```

InputFile 2: Paramotopy input file demonstrating use of computer-generated mesh, and constant declaration. Line 1 indicates 7 equations (lines 2-8), in 1 variable group (line 9), with two parameters (lines 17-21), and zero constant declarations. Line 15's `1` indicates Paramotopy should look for the file on the next line, titled `robomc_10000`. The name of the file is arbitrary, but should be at the same path at the input file.

3.1 Monte Carlo Input

4 Options

Persistent Configuration of the Paramotopy program is maintained through the `./paramotopy/prefs.xml` file located in the home directory.

4.1 Parallelism

4.2 Bertini

5 Troubleshooting