Contents

1 Overview **1**

2 Distribution **2**

2.1 What and Where 2

7.6 Travel Time Curves 27

A Installing **28**

A.1 Unix 28

A.2 MacOS 29

A.3 Windows 29

B Troubleshooting **30**

Disclaimer and License

```
The TauP Toolkit: Flexible Seismic Travel-Time and Raypath Utilities.
Copyright (C) 1998-2000 University of South Carolina

This program is free software; you can redistribute it and/or
modify it under the terms of the GNU General Public License
as published by the Free Software Foundation; either version 2
of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program; if not, write to the Free Software


The current version of The TauP Toolkit can be found at
http://www.seis.sc.edu

Bug reports and comments should be directed to
H. Philip Crotwell, crotwell@seis.sc.edu or
Tom Owens, owens@seis.sc.edu
```

The TauP Toolkit is available free under the terms of the GNU General Public License, found in the COPYING file within the distribution. It gives you specific rights to use, modify and redistribute this software. Please be aware of section 2 of the license which specifically prevents you from redistributing this software incorporated, in whole or on part, into to a work, unless that work is also covered by the GNU General Public License. Please see the Free Software Foundation's web site, wwwg50(www)fo50(and) and

1 Overview

The algorithms employed within the TauP package are based on the method of

2 Distribution

2.1 What and Where

The current distribution of the TauP package is 1.1, dated February 9, 2001.

The distribution directory obtained from either the gzipped tar file or the jar file contains:

README	getting started information
exampleProperties	example properties file
HISTORY	change log
COPYING	the GNU GPL license
bin	directory containing "wrapper scripts" to start the tools, each will print a usage with a -help command line argument. The TAUP_HOME environ-

3 Tools

Tools included with the TauP package:

`taup_time`	calculates travel times.
`taup_pierce`	calculates pierce points at model discontinuities and specified depths.
`taup_path`	calculates ray paths, depth versus epicentral distance.
`taup`	a GUI that incorporates the time, pierce and path tools. This requires swing, and hence may not work on some java1.1 systems.
`taup_curve`	calculates travel time curves, time versus epicentral distance.
`taup_table`	outputs travel times for a range on5 Td4241(at)a55(s)-250(Gutrae1963 Tf 0 0 Td[(table)]TJ/F31 9

taup.source.depth initial depth of the source, 0.0 km by default.

taup.phase.list initial phase list, combined with taup.phase.file. The defaults are p, s, P, S, Pn, Sn, PcP, ScS, Pdiff,difPcyPKSn, difPcyPKdiKSn, difPcyPKIKSn, difault.

3.2 TauP_Time

TauP_Time takes a .

```
-pierce depth      -- adds depth for calculating pierce points
-nodiscon          -- only prints pierce points for the depths added with -pierce

-o outfile         -- output is redirected to "outfile"
-debug             -- enable debugging output
-verbose           -- enable verbose output
-version           -- print the version
-help              -- print this out, but you already know that!
```

The `-rev`, `-turn` and `-under` flags are useful for limiting the output to just those points you care about. The `-pierce depth` option allows you to specify a "pierce" depth that does not correspond to an actual discontinuity. For instance, where does a ray pierce 300 kilometers above the CMB?

For example:

```
taup_
```

data file. The output is put in taup_

The user should be very careful about previously set header variables. TauP_

The usage is:

4 Phase naming in TauP

the time returned would actually be for `P406.7s`. The code "taup_time" would note that this had been done. Obviously, care should be taken to ensure that there are no other discontinuities closer than the one of[(don0 -11.955 Td[(in

10. The symbol `kmps` is used to get the travel time for a specific horizontal phase velocity. For example, `2kmps` represents a horizontal phase velocity of 2 kilometers per second. While the calculations for these are trivial,

5.2 Using Saved Tau Models

There are three ways of finding a previously generated model file. First, as a standard model as part of the distribution. Second, a list of directories and jar files to be searched can be specified with the taup.model.path property. Lastly, the path to the actual model file may be specified. TauP searches each of these places in order until it finds a model that matches the name.

1. Standard Model.

   TauP first checks to see if the model name is associated with a standard model. Several standard models are included within the distributed jar file. They include iasp91 (Kennett and Engdahl, 1991), prem (Dziewon-

6 Programming Interface

phase names that can be used in the interactive code can be used here. Also, duplicates are checked for and eliminated before being added. The method signature is
int TauPAppendPhases(TauPStruct taup, char *phaseString) ;

TasesauPCalculate alculates all arrivals for all of the current phases for the distance specified in the second argument. An initialized TauPStruct is passed as the first argument. The method signature is
int TauPCalculate(TauPStruct taup, double degrees) ;

TasesauPGetNumArrivals eturns the number of arrivals found with the last call to TauPCalculate, above. A negative number indicates an error. An initialized TauPStruct is passed as the first argument. The method signature is
int TauPGetNumArrivals(TauPStruct taup) ;

TasesauPGetArrival eturns the ith arrival 0(to)-2094(with)-293(the)-293(last)-293(call)-294(to)-293(T)80(auPCalculate,)-304(abo)15(v)

```
Parameters are:
taup.create.minDeltaP = 0.1 sec / radian
taup.create.maxDeltaP = 8.0 sec / radian
taup.create.maxDepthInterval = 115.0 kilometers
taup.create.maxRangeInterval = 1.75 degrees
taup.create.maxInterpError = 0.03 seconds
taup.create.allowInnerCoreS = true
Slow model time=39714 801 P layers,907 S layers
T model time=7480
Done Saving ./simpleMod.taup
Done!
Done!
piglet 11>ls
simpleMod.nd     simpleMod.taup
```

The file `simpleMod.taup` contains all of the information about the model. This process needs to be done only once for each velocity model. The times appearing in the output are in milliseconds, and do not refle9a-249(neePFarteMod.taup)]TJ 0 -1

```
Enter Distance or Option [hrpclseabmq]: h
Enter Depth: 143.2
Enter Distance or Option [hrpclseabmq]: c
Enter phases (ie P,p,PcP,S): P,S,PcP,ScS,SKS,sS,SS,PKKP
Enter Distance or Option [hrpclseabmq]: 75

Model: simpleMod
```

```
37.30 2891.00
```

A Installing

The installation for TauP under UNIX is quite simple. And with Java's platform independence, the package should be usable on a Mac oblew Gmachin.AIUnix]TJ ET 1 0 0 1 4 -2103.021cm 1 1 1 1 k 1 1 1 1 K 1 0 0 1 -12.4530 cm 1T /F3219.963 Tf 0 (

```
${TAUP_HOME}/bin
```

B Troubleshooting

There are a few idiosyncrocies about the codes and Java in general that you may run into.

1.