Package ‘TauP.R’

July 2, 2014

Type Package

Title Earthquake Traveltime Calculations for 1-D Earth Models

Version 1.1

Date 2012-03-19

Author Jake Anderson, Jonathan Lees; largely translated from the TTBOX Matlab toolbox by Martin Knapmeyer (http://www.dr-knapmeyer.de/downloads/)

Maintainer Jake Anderson <ajakef@gmail.com>

Description Evaluates traveltimes and ray paths using predefined Earth (or other planet) models. Includes phase plotting routines. The IASP91 and AK135 Earth models are included, and most important arrival phases can be evaluated.

Suggests RSEIS

License GPL

LazyLoad yes

Repository CRAN

Date/Publication 2012-08-13 07:48:15

NeedsCompilation no

R topics documented:

`TauP.R-package` *Teleseismic Ray Tracing and Arrival Times*

Description

This package facilitates calculation of travel distances and times of global seismic phases using 1-D planet models. Preset Earth models are included, but users may create their own or use other models (for Earth, or any planet/moon). Basic graphing functions are included. This package has been validated using the Java TauP package (Crotwell et al, 1999.)

Details

Package:	TauP.R
Type:	Package
Version:	1.0
Date:	2010-12-16
License:	GPL
LazyLoad:	yes

Note

This package is based on Martin Knapmeyer's TTBOX package for MATLAB (2007 release, available at http://www.dr-knapmeyer.de/downloads/), and much credit is owed to him for writing this original toolbox. Many substantial changes have been made in order to improve efficiency, and more are planned for future releases.

Author(s)

Jake Anderson, Jonathan Lees

References

M Knapmeyer. TTBox: A MatLab Toolbox for the Computation of 1D Teleseismic Travel Times. Seismological Research Letters; November/December 2004; v. 75; no. 6; p. 726-733; DOI: 10.1785/gssrl.75.6.726

Crotwell, H. P., T. J. Owens, and J. Ritsema (1999). The TauP Toolkit: Flexible seismic travel-time and ray-path utilities, Seismological Research Letters 70, 154 160.

See Also

RSEIS, GEOmap

Examples

```
data(model)

Rayfan('P', 500, model)

Traveltime('SKKS', 200, 10, model)
```

ak135 *ak135 Earth Model*

Description

Planet model using the data from the ak135 1-D model.

Usage

```
data(ak135)
```

Format

List with following elements:

- **z** Sample depths (km)
- **vp** Sample P wave velocities (km/s)
- **vs** Sample S wave velocities (km/s)
- **rho** Sample densities (kg/m^3)
- **qp** P attenuation
- **qs** S attenuation
- **name** Model name
- **rp** Planet radius
- **year** Year published
- **conr** Depth to Conrad (upper crust/lower crust) discontinuity
- **moho** Depth to Mohorovicic (top of mantle) discontinuity
- **d410** Depth to top of transition zone
- **d520** Depth to olivine beta-gamma transition
- **d660** Depth to top of lower mantle
- **cmb** Depth to core-mantle boundary
- **icb** Depth to inner core boundary

References

Kennett, B.L.N. Engdahl, E.R. & Buland R., 1995. Constraints on seismic velocities in the Earth from travel times, Geophys J Int, 122, 108-124

Examples

```
data(ak135)

Earthplot(ak135)

Traveltime('P', 60, 0, ak135)
```

AnalyzeLVZ *Analyze Low Velocity Zones*

Description

Identifies low velocity zones and improves sampling to allow more accurate raypath calculation.

Usage

```
AnalyzeLVZ(v, vsec, z, rp)
```

Arguments

`v`	Velocity vector (km/s)
`vsec`	Other velocity vector (km/s)
`z`	Depth vector (km)
`rp`	Planet radius

Details

Only v is checked for LVZs. However, since a velocity profile requires both P and S velocities, the other velocity vector is provided as vsec and interpolated within LVZs found in v.

Interpolated velocities might not match those returned by InterpModel because calculations are done after a flat earth transform here.

Value

List with following elements:

`newv`	Velocity (of the same type as input v) vector at new depths (km/s)
`newvsec`	Velocity (of the same type as input vsec) vector at new depths (km/s)
`newz`	New depths sampled (km)
`criticalz`	Critical depths requiring special treatment (km)

Author(s)

Jake Anderson

Examples

```
data(model)
v = model$vp
vsec = model$vs
z = model$z
rp = model$rp

AnalyzeLVZ(v, vsec, z, rp)
```

`CalcTP` *Calculate Layer Traveltime*

Description

Calculates the traveltime through a single layer.

Usage

```
CalcTP(p, v, z, zmin, zmax, novertex = 0)
```

Arguments

`p`	Ray Parameter (s/deg)
`v`	Velocities at top and bottom of layer (km/s)
`z`	Depth at top and bottom of layer (km)
`zmin`	Minimum allowed depth in layer (km)
`zmax`	Maxiumum allowed depth in layer (km)
`novertex`	Optional: if TRUE, vertex cannot be found in layer

Details

Regrettably, this routine is not vectorized. This will be corrected in later versions. This is a subordinate routine to CalcTPsum.

Value

Traveltime between zmin and zmax (s).

Author(s)

Jake Anderson

Examples

```
### Can only be called from CalcTPsum
```

CalcTPsum *Calculate Traveltime along Ray Leg*

Description

Wrapper for CalcTP to calculate a traveltime over many layers.

Usage

```
CalcTPsum(p, v, z, zmin, zmax, novertex)
```

Arguments

p	Ray parameter (s/deg)
v	Velocity vector (km/s)
z	Depth vector (km)
zmin	Minimum depth (km)
zmax	Maximum depth (km)
novertex	Flag to prevent handling of vertices

Details

Note that all depths and velocities provided here are in flat earth coordinates. This is a subordinate routine for FindTime4p and is not intended for human use.

Value

Traveltime along ray leg (s).

Author(s)

Jake Anderson

Examples

```
##### Subordinate routine
```

`CalcXP` *Calculate Horizontal Travel*

Description

Calculates horizontal travel within a single layer.

Usage

```
CalcXP(p, v, z, zmin, zmax, novertex)
```

Arguments

`p`	Ray Parameter (s/deg)
`v`	Velocity at top and bottom of layer (km/s)
`z`	Depth at top and bottom of layer (km)
`zmin`	Minimum allowed depth (km)
`zmax`	Maximum allowed depth (km)
`novertex`	Block handling of vertices if TRUE.

Details

All depths and velocities must be flat earth transformed. This is a subordinate routine for CalcXPsum. Regrettably, this is not vectorized; this will be corrected in later editions.

Value

Horizontal distance traveled in layer in flat earth coordinates.

Author(s)

Jake Anderson

Examples

```
#### Not a user routine: subordinate to CalcXPsum.
```

`CalcXPsum` *Horizontal Distance along Ray Leg*

Description

Calculates horizontal distance traveled by ray over given depth range.

Usage

```
CalcXPsum(p, v, z, zmin, zmax, novertex)
```

Arguments

`p`	Ray parameter (s/deg)
`v`	Velocity vector (km/s)
`z`	Depth vector (km)
`zmin`	Minimum depth (km)
`zmax`	Maximum depth (km)
`novertex`	Flag to prevent consideration of vertices

Details

All depths and velocities are flat earth coordinates. This routine is not vectorized; vectorization is a high priority for future releases. This routine is subordinate to FindDist4p.

Value

Horizontal travel distance between zmin and zmax (km, flat earth).

Author(s)

Jake Anderson

Examples

```
### Not a user routine--subordinate to FindDist4p.
```

ConvAng2p *Angle to Ray Parameter*

Description

Convert between ray angle (from vertical) and ray parameter.

Usage

```
ConvAng2p(phase, h, angle, model = NULL, vp = NULL, vs = NULL, rp =
NULL)
ConvP2Ang(phase, h, p, model = NULL, vp = NULL, vs = NULL, rp = NULL)
```

Arguments

phase	Arrival phase (e.g. 'P' or 'SKS')
h	Depth (km) at which to convert.
angle	Takeoff angle (degrees). 0 is downward, 180 is upward
p	Ray parameter (s/deg)
model	Planet model
vp	P wave velocity at depth h (km/s)
vs	S wave velocity at depth h (km/s)
rp	Planet radius (km)

Details

Either 'model' or all of 'vp', 'vs', 'rp' must be provided. p and angle may be vectors; other arguments may not.

Value

For ConvAng2p, returns a vector of ray parameters (s/deg) corresponding to values in 'angle'.

For ConvP2Ang, returns a vector twice the length of 'p', with all upward angles corresponding to 'p' followed by all downward angles.

Author(s)

Jake Anderson

Examples

```
data(model)
ConvP2Ang('P',100,1,model)

ConvAng2p('P',100,30,model)
```

ConvP2Vdepthinv *Vertex Depth and Ray Parameter*

Description

Calculate vertex depth given ray parameter or vice-versa.

Usage

```
ConvP2Vdepth(p, v, r, h, rp, discons)
ConvP2Vdepthinv(rpd, v, r)
```

Arguments

rpd	Ray vertex radius (km)
v	Planet velocity structure (km/s)
r	Radii corresponding to v
p	Ray parameter (s/deg)
h	Focal radius (km)
rp	Planet radius (km)
discons	Vector of discontinuity radii (km, from FindDiscon)

Details

Note that these functions use radii, not depths, so h would be 6371 (or whatever planet radius you're using) - focal depth.

Value

ConvP2Vdepth: Radius of ray vertex (km)

ConvP2Vdepthinv: Ray parameter (s/deg)

Author(s)

Jake Anderson

Examples

```
data(model)

ConvP2Vdepth(7, model$vp, 6371 - model$z, 6361, 6371, FindDiscon(model))

ConvP2Vdepthinv(4881.467, model$vp, 6371 - model$z)
```

`ConvVdepth2p` *Vertex Depth to Ray Parameter*

Description

Calculates ray parameter given the vertex depth of a ray.

Usage

```
ConvVdepth2p(model, z)
```

Arguments

`model`	planet model
`z`	Vertex depth (km)

Value

A list with the following elements:

`prayp`	P wave ray parameter
`srayp`	S wave ray parameter
`newz`	Vertex depth

Author(s)

Jake Anderson

See Also

ConvP2Vdepth, ConvP2Vdepthinv

Examples

```
data(model)

ConvVdepth2p(model, 300) # calculates p for a ray bottoming at 300 km
```

`DistSummary` *Arrival Summary*

Description

Determine arrival times and information for all major phases arriving at a certain epicentral distance, and plot ray trajectories.

Usage

```
DistSummary(delta, h, model, phaselist = 'default', prop = "vp", image.col = heat.colors(500), n = 200
```

Arguments

`delta`	Epicentral distance (degrees)
`h`	Focal depth (km)
`model`	Planet model
`phaselist`	Either 'default' for all available phases, or a character vector including desired phases
`prop`	Property by which to scale planet image: one of 'vp', 'vs', 'rho'.
`image.col`	Vector of colors for image
`n`	Resolution of image (pixels per side)
`...`	Other parameters for Rayfan

Details

This function is really just a wrapper for Rayfan to calculate arrivals for many phases at just one epicentral distance. Since each phase must be calculated separately, the use of the default phaselist will result in a long calculation time (minutes), and the plot will probably be crowded. It is generally better to define phaselist as a smaller vector or use Rayfan instead.

Value

Returns a list with the following elements:

`p`	Ray parameter for each arrival
`t`	Travel time for each arrival
`dist`	Epicentral distance (should be approximately the input dist)
`phase`	Phase of each arrival

Author(s)

Jake Anderson

See Also

Rayfan, Traveltime, Earthplot

Examples

```
data(model)

# for an event occurring 100 degrees away at a depth of 40 km:

DistSummary(delta = 100, h = 40, model = model)
```

Earthplot *Planet Cross-section*

Description

Plots a planet cross-section for a specified model.

Usage

```
Earthplot(model, prop = "vp", image.col = heat.colors(500), n = 200, add = FALSE, ...)
```

Arguments

model	Planet model
prop	Property to scale image by: one of 'vp', 'vs', 'rho'
image.col	Color vector for the image
n	Number of pixels per side of the plot
add	Add to existing figure? 'image' overplots whatever is below it, so rarely useful.
...	Other parameters for 'image'

Details

Plots lines illustrating discontinuities with background colors indicating one of vp, vs, or density.

Value

None, plots only.

Author(s)

Jake Anderson

See Also

Rayfan, DistSummary

Examples

```
data(model)

Earthplot(model)
```

`EmptyModel` *Empty Planet Model*

Description

Create an empty planet model with defined, named elements including NaN or length 0 values.

Usage

```
EmptyModel()
```

Value

Planet model containing no information.

Author(s)

Jake Anderson

Examples

```
EmptyModel()
```

`FindDiscon` *Identify Discontinuities*

Description

Identify discontinuities in planet model.

Usage

```
FindDiscon(model)
```

Arguments

`model`	Planet model

Details

Note that this returns radii, not depths!

Value

Vector of discontinuity radii (km)

Author(s)

Jake Anderson

Examples

```
data(model)
FindDiscon(model)
```

FindDist4p *Epicentral Distance*

Description

Calculates epicentral distance given focal depth and ray parameter or takeoff angle

Usage

```
FindDist4p(phase, h, model, p, takeoff)
```

Arguments

phase	Phase of arrival (e.g. 'P', 'pS')
h	Focal depth (km)
model	Planet model
p	Ray parameter (s/deg)
takeoff	Takeoff angle (deg)

Details

Only one of 'p', 'takeoff' needs to be specified, and may be a vector. 'phase' and 'h' must be scalars.

Value

List including the following elements:

dist	Vector of surface distances traveled (deg), corresponding to the values in 'p' or 'takeoff'
segx	List of vectors corresponding to 'p' or 'takeoff'. Each vector includes distance coordinates (deg) along the ray path.
segz	List of vectors corresponding to 'p' or 'takeoff'. Each vector includes depth coordinates (km) along the ray path.

`segtyp`	List of vectors corresponding to 'p' or 'takeoff'. Each vector includes wave type ('P' or 'S') for each segment in the ray. Note that vectors in 'segtyp' have one fewer element than vectors in 'segx' and 'segz' because they describe segments, not points.
`resp`	Vector of ray parameters for each ray (s/deg).

Author(s)

Jake Anderson

See Also

Traveltime, FindTime4p

Examples

```
data(model)

FindDist4p('SKKS',100,model,c(4,5))
```

`FindP4Dist` *Ray Parameter for Epicentral Distance*

Description

Calculates ray parameter and takeoff angle to reach given epicentral distances. Including a pscan improves speed if you already have it, but is not necessary.

Usage

```
FindP4Dist(phase, deltalist, h, model, pscan = NULL)
```

Arguments

`phase`	Wave arrival phase (e.g. 'P', 'SKS')
`deltalist`	Vector of epicentral distances (degrees)
`h`	Focal depth (km)
`model`	Planet model
`pscan`	Output of MakePscan

Value

List with following values:

`p`	Vector of ray parameters (s/deg)
`a`	Vector of takeoff angles (deg)
`d`	Vector of corresponding epicentral distances (deg)
`deltain`	Vector of target epicentral distances (deg)

Author(s)

Jake Anderson

Examples

```
data(model)
FindP4Dist('P', 60, 100, model)
```

FindPrange *Ray Parameter Range for Phase*

Description

Determine window of possible ray parameters for given phase.

Usage

```
FindPrange(phase, imodel, h, dangle)
```

Arguments

phase	Wave arrival phase (e.g., 'P' or 'ScS')
imodel	Planet model (improved by ImproveModel if possible)
h	Focal depth (km)
dangle	Angle resolution of output (deg)

Value

List with following elements:

angles	Vector of takeoff angles spaced 'dangle' apart in acceptable range (deg)
minangle	Minimum takeoff angle for 'phase'
maxangle	Maximum takeoff angle for 'phase'

Author(s)

Jake Anderson

Examples

```
data(model)
imodel = ImproveModel(model)$newmodel
FindPrange('P', imodel, 100, 10)
```

`FindRoots` *Find Roots of X(a) Error Function*

Description

Finds solutions for epicentral distance error - takeoff angle function.

Usage

```
FindRoots(phase, delta, h, model, startalpha, startdist)
```

Arguments

`phase`	Wave arrival phase (e.g. 'P', 'S'.)
`delta`	Epicentral distance (degrees)
`h`	Focal depth (km)
`model`	Planet model
`startalpha`	Takeoff angle interval containing root (degrees)
`startdist`	Epicentral distance interval containing root (degrees)

Value

List with the following elements:

`p`	Solution ray parameter (s/deg)
`a`	Solution takeoff angle (deg)
`d`	Solution epicentral distance (deg)

Author(s)

Jake Anderson

Examples

```
data(model)
phase = 'P'
delta = 60
h = 100
startalpha = c(30, 31)
startdelta = FindDist4p('P', 100, model, takeoff = startalpha)$dist

FindRoots(phase, delta, h, model, startalpha, startdelta)
```

FindTime4p *Travel time for Ray Parameter*

Description

Calculates a travel time given a phase, focal depth, model, and ray parameter.

Usage

```
FindTime4p(phase, h, p, model, anglemode = "rayparm", takeoff = NULL)
```

Arguments

phase	Arrival phase (e.g. 'P', 'SKS')
h	Focal depth (km)
p	Ray Parameter (s/deg)
model	Planet model
anglemode	One of 'rayparm' (if the input ray parameter is to be used) or 'angle' (if the input takeoff angle is to be used)
takeoff	Takeoff angle (deg)

Details

'takeoff' and 'p' must be scalars–unlike many of the other functions provided, FindTime4p is not vectorized.

Value

tt	Phase travel time (s)
vdep	Vertex radius (km)
resp	Ray parameter (s/deg)

Author(s)

Jake Anderson

See Also

Traveltime, FindDist4pn

Examples

```
data(model)

FindTime4p('P', 100, 6, model)

FindTime4p('P', 100, NaN, model, anglemode = 'angle', 40)
```

GreatDist — *Distance Along Great Circle Arc*

Description

Distance Along Great Circle Arc in degrees, kilometers

Usage

```
GreatDist(LON1, LAT1, LON2, LAT2, EARTHRAD= 6371)
```

Arguments

`LON1`	Longitude, point1
`LAT1`	Latitude, point1
`LON2`	Longitude, point2
`LAT2`	Latitude, point2
`EARTHRAD`	optional earth radius, default = 6371

Value

LIST:

`drad`	distance in radians
`ddeg`	distance in degrees
`dkm`	distance in kilometers

Author(s)

Jonathan M. Lees <jonathan.lees@unc.edu>

Examples

```
##########   get distance between London, England and Santiago, Chile
london = c(51.53333, -0.08333333)
santiago = c(-33.46667, -70.75)

GreatDist(london[2], london[1], santiago[2], santiago[1])
```

HoneP *Hone Ray Parameter*

Description

Refines ray parameter to help correct numerical inaccuracies. The indicated phase exists for the output ray parameter, but might not for the input.

Usage

```
HoneP(oldp, oldangle, direction, phase, h, model)
```

Arguments

`oldp`	Ray parameter to be honed (s/deg)
`oldangle`	Takeoff angle to be honed (deg)
`direction`	Search direction: 'up', 'down', or 'both'
`phase`	Arrival phase: (e.g. 'PP', 'SKS')
`h`	Focal depth
`model`	Planet model

Value

`newp`	Correct ray parameter or NaN
`newangle`	Correct takeoff angle or NaN

Author(s)

Jake Anderson

Examples

```
### not a user routine
```

`iasp91` *IASP91 Earth Model*

Description

Planet model using the data from the IASP91 1-D model.

Usage

```
data(iasp91)
```

Format

List with following elements:

- **z** Sample depths (km)
- **vp** Sample P wave velocities (km/s)
- **vs** Sample S wave velocities (km/s)
- **rho** Sample densities (kg/m^3)
- **qp** P attenuation
- **qs** S attenuation
- **name** Model name
- **rp** Planet radius
- **year** Year published
- **conr** Depth to Conrad (upper crust/lower crust) discontinuity
- **moho** Depth to Mohorovicic (top of mantle) discontinuity
- **d410** Depth to top of transition zone
- **d520** Depth to olivine beta-gamma transition
- **d660** Depth to top of lower mantle
- **cmb** Depth to core-mantle boundary
- **icb** Depth to inner core boundary

References

Kennet BLN, Engdahl ER, 1991. Traveltimes for global earthquake location and phase identification. Geophysical Journal International 105(2) 429-465.

Examples

```
data(iasp91)

Earthplot(iasp91)

Traveltime('P', 60, 0, iasp91)
```

ImproveModel *Improve Planet Model*

Description

Increase sampling in model and identify important depths (discontinuities, triplications, LVZs) and corresponding p and s ray parameters.

Usage

```
ImproveModel(oldmodel)
```

Arguments

`oldmodel`	Existing planet model.

Details

The element $criticalrays is added to the output element $newmodel. $criticalrays includes a vector of depths ($z), p ray parameters ($p), and s ray parameters ($s).

Value

List including following elements:

`newmodel`	Improved model, including criticalrays element
`newdepths`	Identified critical depths

Author(s)

Jake Anderson

Examples

```
data(model)
imodel = ImproveModel(model)
```

`InterpModel` *Linear Interpolation of Planet Model*

Description

Interpolates a model at provided depths.

Usage

```
InterpModel(model, newz = NULL, preserve = NULL)
```

Arguments

`model`	Planet model
`newz`	Depths at which to interpolate (km)
`preserve`	If NULL (default), TRUE, or 'preserve', preserve discontinuities in interpolated result

Value

Planetary object variable containing data at the desired depths

Author(s)

Jake Anderson

Examples

```
data(model)
InterpModel(model, 10, preserve = FALSE)
```

`LinInterp` *Linear Interpolation*

Description

Linearly interpolates, allowing multiple y-values for a given x-value.

Usage

```
LinInterp(xin, yin, xout, mode = 'data')
```

Arguments

xin	Input x vector
yin	Input y vector
xout	x-values at which to interpolate
mode	How to handle x-values with multiple y-values: one of 'jump', 'data', 'all'

Details

Regarding the 'mode' argument: 'data' interpolates using the mean of all y-values for the given x-value, while 'jump' or 'all' uses only the y-value on the same side of the discontinuity as the element of 'xout'.

Value

Vector of interpolated y-values corresponding to xout.

Author(s)

Jake Anderson

Examples

```
xin = c(1, 2, 3, 3, 4, 5)
yin = c(0, 0, 0, 1, 1, 1)
xout = 3.5

LinInterp(xin, yin, xout, 'all')
LinInterp(xin, yin, xout, 'data')
```

LVZSmp *Identify LVZs*

Description

Identify low velocity zones in a planet model and improve depth sampling in them.

Usage

```
LVZSmp(oldmodel)
```

Arguments

oldmodel	Planet model

Value

List with following elements:

`lvzextra`	Planet model only containing addtional depth samples to improve model.
`criticalz`	Depths to bottoms of LVZs (km)

Author(s)

Jake Anderson

Examples

```
data(model)
LVZSmp(model)
```

`MakePscan` *Find Distance of p Function*

Description

Constructs a distance for ray parameter function for the range of relevant ray parameters for a given phase.

Usage

```
MakePscan(phase, h, imodel)
```

Arguments

`phase`	Earthquake wave arrival phase (e.g. 'P', 'SKKS')
`h`	Focal depth (km)
`imodel`	Planet model returned by ImproveModel

Value

List with following elements:

`phase`	Arrival phase
`h`	Focal depth (km)
`angles`	Takeoff angles (degrees)
`p`	Corresponding ray parameters (s/deg)
`dist`	Corresponding epicentral distances (degrees)
`vp`	P wave velocity at focus
`vs`	S wave velocity at focus
`starts`	Starting indices of intervals (1:(length(p) - 1))
`ends`	Ending indices of intervals (2:length(p))

Author(s)

Jake Anderson

Examples

```
data(model)

phase = 'P'
h = 100
imodel = ImproveModel(model)$newmodel

MakePscan(phase, h, imodel)
```

---

meshgrid *Create a mesh grid like in Matlab*

Description

Creates 2D matrices for accessing images and 2D matrices

Usage

```
meshgrid(a, b)
```

Arguments

a	x vector components
b	y vector components

Details

returns outer product of x-compnents and y-components for use as index arrays

Value

x	length(y) by length(x) matrix of x indicies
y	length(y) by length(x) matrix of y indicies

Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

Examples

```
meshgrid(1:5, 1:3)
```

`model` *ak135 Earth Model*

Description

Planet model using the data from the ak135 1-D model.

Usage

```
data(model)
```

Format

List with following elements:

- **z** Sample depths (km)
- **vp** Sample P wave velocities (km/s)
- **vs** Sample S wave velocities (km/s)
- **rho** Sample densities (kg/m^3)
- **qp** P attenuation
- **qs** S attenuation
- **name** Model name
- **rp** Planet radius
- **year** Year published
- **conr** Depth to Conrad (upper crust/lower crust) discontinuity
- **moho** Depth to Mohorovicic (top of mantle) discontinuity
- **d410** Depth to top of transition zone
- **d520** Depth to olivine beta-gamma transition
- **d660** Depth to top of lower mantle
- **cmb** Depth to core-mantle boundary
- **icb** Depth to inner core boundary

References

Kennett, B.L.N. Engdahl, E.R. & Buland R., 1995. Constraints on seismic velocities in the Earth from travel times, Geophys J Int, 122, 108-124

Examples

```
data(model)

Earthplot(model)

Traveltime('P', 60, 0, model)
```

OptimizeDist *Find Extrema in D(a)*

Description

Engine routine that identifies local extrema in the D(a) (epicentral distance/takeoff angle) function.

Usage

```
OptimizeDist(alphalimit, deltalimit, phase, h, imodel)
```

Arguments

`alphalimit`	Angle interval (2-element vector, deg)
`deltalimit`	Epicentral distances of alphalimit
`phase`	Arrival phase (e.g. 'P', 'PKIKP')
`h`	Focal depth (km)
`imodel`	Improved planet model (from ImproveModel)

Details

OptimizeDist assumes that D(a) has only one extremum over the interval, and is finite and defined everywhere. It uses a Golden Section Search algorithm to find the extremum.

Value

`extremalpha`	Takeoff angle for identified extreme epicentral distance (s/deg)
`extremp`	Ray parameter for extremalpha (s/deg)
`extremdelta`	Identified extreme epicentral distance

Author(s)

Jake Anderson

Examples

```
### not a user routine
```

`polaraxis` *Polar Plot Axis*

Description

Writes a circular 'theta' axis around a polar plot.

Usage

```
polaraxis(rp = 6371, at = 0:17 * 20)
```

Arguments

`rp`	Plot radius
`at`	Angles to label (degrees)

Value

None; graphical side effects only.

Author(s)

Jake Anderson

See Also

PolarPlot

Examples

```
# Borrowed from Earthplot:

par(mar = c(1.1,1.1,4.1,1.1))
plot(0,type='n',xlim = 1.15 * c(-6271, 6371),ann=FALSE,axes=FALSE,asp=1)

PolarPlot(0:360,6371,type='l',method=lines,degree=TRUE,geographical=TRUE,col='black')

polaraxis(6371)
```

PolarPlot *Polar Plot*

Description

Plot polar coordinates

Usage

```
PolarPlot(theta, r, degrees = FALSE, method = plot, geographical = FALSE, ...)
```

Arguments

theta	Angle coordinates
r	Radius coordinates
degrees	Logical: is 'theta' in degrees?
method	Plot method: can be plot, lines, or points. Note that it expects function names, not character strings.
geographical	Logical: if TRUE, 'theta' goes clockwise from cartesian (0,1) rather than counterclockwise from cartesian (1,0)
...	Other plotting parameters

Value

None; graphical side effects only.

Author(s)

Jake Anderson

Examples

```
PolarPlot(pi/8 * 1:16, 0:15, method = plot)
```

Rayfan *Ray Fan*

Description

Calculate travel times and plot ray trajectories of phase(s) from focus to receiver(s).

Usage

```
Rayfan(phaselist, h, model, deltalist = 1:17 * 20, minp = 0.5, plot = TRUE, add = TRUE, col = rep("blac
```

Arguments

`phaselist`	Character vector of phases to plot (e.g. c('P','S','PP','SS'))
`h`	Focal depth (km)
`model`	Planet model
`deltalist`	Vector of epicentral distances (degrees)
`minp`	Smallest allowed ray parameter (s/deg) to prevent errors near the center of the planet.
`plot`	Logical: plot ray trajectories?
`add`	Add to existing Earthplot/Rayfan figure?
`col`	Color vector for 'image'
`verbose`	Print information as calculations are done?
`mirror`	Logical: should delta = x be considered equivalent to delta = 360 - x?

Details

It is useful to remember phases like PKKP that travel more than 180 degrees may physically arrive in the same place as a phase that travels less than 180 degrees like PKP, but this package does not recognize it unless 'mirror' is TRUE.

Value

Output from each Traveltime calculation is concatenated into the following list:

`tt`	vector of traveltimes (s)
`p`	vector of ray parameters (s/deg)
`angles`	vector of takeoff angles (degrees)
`dists`	vector of epicentral distances (degrees)

Author(s)

Jake Anderson

See Also

Earthplot, Traveltime, DistSummary

Examples

```
data(model)

Rayfan(c('S','ScS'),100,model)
```

ReadND *Read Model File*

Description

Scans a model from .nd or .clr format into R.

Usage

```
ReadND(filename, verbose = TRUE)
ReadCLR(filename, z = 'default')
```

Arguments

`filename`	Filename, including path
`verbose`	Logical: should details be printed during run?
`z`	Vector of depths at which velocities should be calculated, or 'default' for a default vector.

Details

.nd refers to 'Named Discontinuity' files (Davis and Henson, 1993), in which properties are provided at each sampled depth. .clr refers to 'Continuous Layer Representation' files (Knapmeyer, 2004), in which coefficients of polynomial approximations of velocities are given for each of several layers.

Value

Planet model corresponding to the .nd/.clr file.

Author(s)

Jake Anderson

References

Knapmeyer, M (2004). TTBox: A MatLab Toolbox for the Computation of 1D Teleseismic Travel Times. Seismological Research Letters, v. 75, no. 6, p. 727-733, DOI 10.1785/gssrl.75.6.726.

Davis, J. P and I. H. Henson (1993). User's Guide to Xgbm: An X-Windows System to Compute Gaussian Beam Synthetic Seismograms (1.1 edition), Alexandria, VA: Teledyne Geotech, Alexandria Laboratories.

Examples

```
## Not run:
model1 = ReadND('somemodel.nd')
model2 = ReadCLR('somemodel.clr', z = seq(from = 0, to = 6371, by = 40) )

## End(Not run)
```

`SlopeInt` *Find Slope and Intercept*

Description

Calculates slope and y-intercept of the velocity-depth function for a layer.

Usage

```
SlopeInt(v, z)
```

Arguments

v	2-element vector of velocities (km/s)
z	2-element vector of depths (km)

Value

List with the following elements:

g	Gradient of velocity-depth linear approximation (km/s / km)
v0	Constant term of velocity-depth linear approximation (km/s)

Author(s)

Jake Anderson

Examples

```
SlopeInt(c(5, 5.1), c(20, 22))
```

StripRepetitions *Remove Repetitions from Phase*

Description

Removes numbers indicating multiples from phase name and lists them separately.

Usage

```
StripRepetitions(phase)
```

Arguments

phase	Wave arrival phase (e.g. 'P', 'SKS2')

Value

List including remaining (unrepeated) phase and number of repetitions.

Author(s)

Jake Anderson

Examples

```
StripRepetitions('PKP5')
```

TransformF2Sz *Flat Earth Transformation*

Description

Transform Flat Earth depth/velocity/distance/ray parameter to Round Earth, and vice-versa.

Usage

```
TransformF2Sz(vf, zf, rp)
TransformS2Fz(vs, zs, rp)
TransformS2Fp(ps, rp)
TransformF2Sdist(xf, rp)
```

Arguments

`vf`	Flat-Earth velocity (km/s)
`zf`	Flat-Earth depth (km)
`rp`	Planet radius (km)
`vs`	Round-Earth velocity (km/s)
`zs`	Round-Earth depth (km)
`ps`	Round-Earth ray parameter (s/deg)
`xf`	Flat-Earth horizontal distance (km)

Value

TransformF2Sz:

`vs`	Round-Earth velocity (km/s)
`zs`	Round-Earth depth (km)

TransformS2Fz:

`vf`	Flat-Earth velocity (km/s)
`zf`	Flat-Earth depth(km)

TransformS2Fp: Flat-Earth ray parameter (s/km)

TransformF2Sdist: Round-Earth surface distance (deg)

Author(s)

Jake Anderson

Examples

```
TransformF2Sz(19, 2700, 6371)

TransformS2Fz(12.5, 2800, 6371)

TransformS2Fp(10, 6371)

TransformF2Sdist(10000, 6371)
```

`Traveltime` *Earthquake traveltimes*

Description

Calculates traveltimes between focus and receiver(s).

Usage

```
Traveltime(phase, delta, h, model, pscan = NULL)
```

Arguments

`phase`	Phase of arrival (such as 'P', 'SKKS', 'PKIKP', etc.)
`delta`	Epicentral distance (degrees)
`h`	Focal Depth (km)
`model`	Planet model
`pscan`	Optional: pscan produced by MakePscan.

Details

Only a single phase, h, and model may be provided, but delta may be a vector. Providing pscan can save considerable calculation time, but is specific to each phase/depth combination, so it's not commonly available.

Value

List with the following elements:

`tt`	vector of traveltimes (s)
`p`	vector of ray parameters (s/deg)
`angles`	vector of takeoff angles (degrees)
`dists`	vector of epicentral distances (degrees)

Author(s)

Jake Anderson

See Also

Rayfan, DistSummary, FindDist4p, FindTime4p

Examples

```
data(model)

delta = seq(from = 30, to = 90, by = 20)
Traveltime('S', delta, 20, model)
```

Index