Package ‘spsh’

March 24, 2019

Type Package

Title Estimation and Prediction of Parameters of Various Soil Hydraulic Property Models

Version 1.0.4

Date 2019-03-19

Author Tobias KD Weber [aut, cre], Efstathios Diamantopoulos [ctb], Mélanie Weynants [ctb]

Maintainer Tobias KD Weber <tobias.weber@uni-hohenheim.de>

Description Estimates model parameters of soil hydraulic property functions by inverting measured data. A wide range of hydraulic models, weighting schemes, global optimization algorithms, Markov chain Monte Carlo samplers, and extended statistical analyses of results are provided.


All models can be extended to account for non-capillary water storage and transport. The isothermal vapour conductivity (Saito, H., Simunek, J. and Mohanty, B.P. (2006) <doi:10.2136/vzj2006.0007>) is calculated based on volumetric air space and


Moldrup, P., Olesen, T., Yoshikawa, S., Komatsu, T., and Rolston, D.E. (2005) <doi:10.1007/01.ss.000196768.44165.1f>,


License  GPL (>= 2)

Encoding  UTF-8

RoxygenNote  6.1.0

LazyData  true

Depends DEoptim (>= 2.2.4), lhs (>= 0.16.0), pracma (>= 2.1.4), FME (>= 1.3.5)

NeedsCompilation  no

Repository  CRAN

Date/Publication  2019-03-24 17:10:03 UTC

R topics documented:

spsh-package ............................................................... 3
gofFun ................................................................. 5
inipopFun ............................................................. 7
KvapFun ............................................................... 8
logLikFun.norm ......................................................... 10
numMualem ............................................................. 12
ptf.cW ................................................................. 13
ptf.vG2FM .............................................................. 14
resFun ................................................................. 16
shpdata1 ............................................................... 18
shypEstFun ............................................................ 20
shypFun ............................................................... 27
shypFun.01110 .......................................................... 31
shypFun.01210 .......................................................... 32
shypFun.01310 .......................................................... 34
shypFun.02110 .......................................................... 35
shypFun.03110 .......................................................... 37
sncFun ............................................................... 38
transBoundFun ......................................................... 39
transFun ............................................................... 40
weightFun ............................................................. 42

Index  44
Description

This package provides a multitude of spsh models, in particular the soil hydraulic property (SHP) models; i.e. the soil water retention curve (WRC) and unsaturated hydraulic conductivity curve (HCC). For the WRC and HCC, (weighted) parameter estimation options exist. Global optimisation is done with the Differential Evolution algorithm `DEoptim`. A number of goodness-of-fit metrics are implemented, some of which routed in Bayesian theory. Also the `DRAM` Markov-chain Monte Carlo algorithm as implemented in `modmcmc`, can be used to sample the posterior. A pedotransfer function to predict soil hydraulic property model parameters based on soil textural information is included.

The analyses of the goodness-of-fit is based on the best parameter estimate. Note, that non-uniform model priors can currently not be specified.

Details

Package: spsh
Type: Package
Version: 1.0.4
Date: 2019-03-19
License: GPL-2

Supported Soil Hydraulic Property Models

<table>
<thead>
<tr>
<th>Code</th>
<th>Model Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>01110</td>
<td>unimodel van Genuchten-Mualem model, with the constraint of $m = 1/n_i$ (van Genuchten, 1980)</td>
</tr>
<tr>
<td>01210</td>
<td>bimodel van Genuchten-Mualem model, with the constraint of $m_i = 1 - 1/n_i$ (Durner, 1994)</td>
</tr>
<tr>
<td>01310</td>
<td>trimodal van Genuchten-Mualem model, with the constraint of $m_i = 1 - 1/n_i$ (Durner, 1994)</td>
</tr>
<tr>
<td>02110</td>
<td>unimodal 2 parametric Kosugi-Mualem model (Kosugi, 1996)</td>
</tr>
<tr>
<td>03110</td>
<td>unimodal base Fredlund-Xing-Mualem model, with the constraint of $m = 1-1/n$ (Fredlund and Xing, 1994)</td>
</tr>
</tbody>
</table>

Framework Model (Brunswick Model)

The Framework Model, as proposed by Weber et al. (2019), can be used with the implemented soil hydraulic property functions.

This option can be accessed by adding "FM" to the code specified in `shpmodel`, e.g. in `shypFun`

Supported Pedo-transfer function

<table>
<thead>
<tr>
<th>Code</th>
<th>Function Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cW</td>
<td>corrected Weynant model of Weynants et al. (2009) and Weihermueller et al. (2017)</td>
</tr>
</tbody>
</table>

(For a mathematical description refer to original literature in the references or the provided R code)
Author(s)

Tobias K.D. Weber<toibas.weber@uni-hohenheim.de>

Biogeophysics Department
Institute of Soil Science and Land Evaluation
University of Hohenheim
Germany

References


See Also

shypFun, shypEstFun
Examples

```r
p <- c("thr" = 0.1, "ths" = 0.4, "alfl" = 0.01, "n1" = 2, "Ks" = 100, "tau" = .5)
h <- 10^seq(-2, 6.8, length = 197)
shyp.L <- sh ypFun.01110(p, h)
```

Description

Calculates goodness-of-fit criteria and the likelihood-based Akaike and Bayesian Information Criterion.

Usage

```r
gofFun(phat, shpmodel = "01110", retdata = NULL, condata = NULL,
       weight, psel, ivap.query = NULL, hclip.query = FALSE)
```

Arguments

- `phat`: vector of non-transformed (back-transformed) model parameters after estimation, i.e. the best fit or maximum likelihood estimate.
- `shpmodel`: Character specifying the soil hydraulic property model.
- `retdata`: Dataframe or matrix with 2 columns. The first with pressure head values in log10 [cm], i.e. pF values, and the second with volumetric water contents in [cm cm⁻³].
- `condata`: Dataframe or matrix with 2 columns. The first with pressure head values in log10 [cm], i.e. pF values, and the second with hydraulic conductivity values log10 [cm d⁻¹].
- `weight`: List of the model residual weights used or estimated by the parameter estimation scheme, to calculate the weighted statistical analyses.
- `psel`: Vector specifying the selected parameters for the parameter estimation from `parL`.
- `ivap.query`: specification of ivap method, if FALSE selected, no isothermal vapour conductivity is considered See Details
- `hclip.query`: Implemented purely for future compatibility. Currently no use. See Details

Details

Statistical analyses of the inverse modelling results
Value

list The output of goffun returns a list of three list, if arguments retdata and condata are both !NULL. Only one corresponding list if only retdata or condata are given as arguments.

The goodness-of-fit and information criteria output calculated on the (weighted) errors are for retdata and condata are:

th list with goodness-of-fit statistics for the retention curve with elements:

- me mean (weighted) error
- mae mean absolute (weighted) error
- mse mean squared (weighted) error
- rss sum of squared (weighted) errors
- rmse root mean squared (weighted) error
- AIC Akaike Information Criteria
- AICc corrected Akaike Information Criteria
- BIC Bayesian Information Criteria
- m number of observations

logKh list with output same as th but for the log10 fitted conductivity curve
combined list with AIC, AICc, and BIC calculated for the multi-objective function if arguments retdata and condata are both !NULL

Author(s)

Tobias KD Weber

References


Examples

data("shpdata1")
retdata <- shpdata1$TS$wrc
condata <- shpdata1$TS$hcc
condata <- condata[!is.na(condata[,1]),]
parL <- list("p" = c("thr" = 0.05, "ths" = 0.45, "alf1" = 0.01, "n" = 2, "Ks" = 100, "tau" = .5),
"psel" = c(1, 1, 0, 1, 1, 1),
"plo" = c(0.001, 0.2, 0.001, 1.1, 1, -2),
"pup" = c(0.3, 0.95, 1, 10, 1e4, 10))
gofL <- goffun(parL$p, shpmodel = "01110", retdata = retdata, condata = condata,
weight = weightfun(weightmethod = "fix1"), parL$psel,
ivap.query = NULL, hclip.query = FALSE)
inipopFun

Generates an Initial Population of Transformed Soil Hydraulic Property Model Parameters

Description

Draws a Latin Hypercube Sample from a set of uniform distributions in the transformed parameter space, in creating a Latin Hypercube Design. This function uses the Columnwise Pairwise (CP) algorithm to generate an optimal design with respect to the S optimality criterion, as implemented in lhs-package.

Usage

```r
inipopFun(p, psel, plo, pup, trans.L, Npop = NA)
```

Arguments

- `p` vector of model parameters
- `psel` vector of selectors.
- `plo` vector of lower boundary values of non-transformed parameters
- `pup` vector of upper boundary values of non-transformed parameters
- `trans.L` list of transformation/backtransformation operators with same length as `p`, `psel`, `plo`, and `pup`.
- `Npop` integer of initial population size

Details

Produces and optimum latin hypercube sample from a bounded uniform distribution.

Value

`n` draws of `k` parameters in an `n x k` Latin Hypercube Sample matrix with values uniformly distributed on user specified bounds.

Author(s)

Tobias KD Weber

See Also

- `optimumLHS`
Examples

## Example based on soil hydraulic property model parameters of `shpmodel = "01110"

```r
# parameters
parL <- list("p" = c("thr" = 0.05, "ths" = 0.45, "alf1" = 0.01, "n" = 2, "Ks" = 100, "tau" = .5),
  "psel" = c(1, 1, 0, 1, 1, 1),
  "plo" = c(0.001, 0.2, 0.001, 1.1, 1, 2),
  "pup" = c(0.3, 0.95, 1, 10, 1e4, 10)
)
# rules for the parameter transformation
ptransfit <- c(function(x)x, function(x)x, log10, function(x)log10(x-1), log10, function(x)x)
# get latin hypercube sample.
test.inipop <- inipopFun(parL$p, parL$psel, parL$plo, parL$pup, ptransfit, Npop = 20)
# plot the latin hypercube
pairs(test.inipop)
```

KvapFun

*Calculates the Isothermal Water Vapour Conductivity.*

**Description**

Calculates the isothermal vapour conductivity as a function of modelled volumetric air content. Different models are implemented enabling the calculation of the relative gas diffusion coefficient \((D_s/D_0)\), based on different expressions for an effective tortuosity.

**Usage**

```r
KvapFun(p, por = p[2], retFun = NA, theta = NA, model = "MQ61",
  Temp = 20, m = 3, pF = seq(-3, 7, length = 501), output = "log10", ...)
```

**Arguments**

- **p**: vector of soil hydraulic property model parameters
- **por**: scalar value giving the fraction of a porous media porosity \([-\ ]\)(value between [0, 1]), defaults to the saturated water content.
- **retFun**: soil hydraulic property function has to be specified if models PMQ, TPM or TPEM are used, necessary to calculate the air content at \(h = 100\) cm for the parameter \(\varepsilon_{100}\).
- **theta**: vector of numerical volumetric water contents [0,1] at which the air content is to be calculated.
model

Implemented models (specify as character):

- B Buckingham (1904)
- P Penman (1940)
- MQ60 Millington and Quirck (1960)
- MQ61 Millington and Quirck (1961)
- GS Grable and Siemer (1968)
- L Lai et al. (1976)
- PMQ Moldrup et al. (1997)
- TPM Moldrup et al. (2004)
- TPEM Moldrup et al. (2005)

Temp Soil temperature [deg C], defaults to 20.

m PMQ model parameter, default m = 3

pF monotonically increasing pF values, defined as log10(|pressure head| cm)

output

Defaults to log10 indicates the isothermal vapour conductivity is returned as log10(conductivity), if output ≠ log10, the output will be in non-transformed values.

Details

More reading on the model is suggested in Weber et al. (2019).

Value

Returns a item vector of isothermal vapour conductivity values, corresponding to the pressure heads defined through argument pF.

Author(s)

Tobias KD Weber

References


logLikFun.norm

### Calculation of the Log-likelihood assuming Identically, independency and Normally Distributed errors

**Description**

Calculates the i-th log-likelihood of each y-hat pair as described in Seber and Wild (2003, ISBN:9780471617600) Eq. 2.46.

**Usage**

```r
logLikFun.norm(y, yhat, sigma)
```
Arguments

y  A vector of \( n \) observed properties/variables of interest.
yhat  A vector of \( n \) model simulated properties/variables of interest.
sigma  A vector of length 1 considering homoscedastic residuals.

Details

The underlying assumption is, that the model residuals (errors) are independently, and identically distributed (i.i.d.) following a normal distribution.

Value

\( \text{log-likelihood} \) value of an normal distribution with \( N(0, \sigma^2) \)

Note

The assumption of i.i.d. and normal distribution is best investigated \textit{a posteriori}.

Author(s)

Tobias K.D. Weber <tobias.weber@uni-hohenheim.de>,

References


Examples

```r
# homoscedastic residuals
sig.s <- .01
y.scat <- rnorm(100, 0, sig.s)

yhat <- (1:100)^1.2
y <- yhat + y.scat
sum(logLikFun.norm(y, yhat, sig.s))
plot(yhat-y)
```
numMualem  

Function to Numerically Compute the Mualem Integral

Description
This function will calculate Mualems integral and returns hydraulic conductivity values.

Usage
numMualem(h, pcon, scap)

Arguments
h  vector of length 1 with pressure head values.
pcon  vector of soil hydraulic conductivity model parameters, the first argument has to be the tortuosity parameter ‘tau’
scap  vector of length 1 of calculated (capillary) saturation values

Details
The numerical solution of Mualems integral relies on the trapezoidal rule of integration.

Value
numMualem  returns a vector of length 1 with calculated conductivity values.

Author(s)
Tobias KD Weber

References

Examples

h <- 10^seq(-3, 6.8, length = 501)
p = c(.05, .5, .01, 1.8, 100, .5)
shyp.L <- shypFun.01110(p, h)

Ks <- p[5]
tau <- p[6]
Se <- shyp.L[['Se']]
Khrnum <- numMualem(h, pcon = tau, scap = Se)

Khnum <- Ks * Se^tau * Khrnum
Corrected Weynants et al. (2009) Pedotransfer Function

Description
This function predicts van Genuchten-Mualem model parameters, setting the residual water content to zero.

Usage
```r
ptf.cW(CLAY, SAND, BD, OC)
```

Arguments
- **CLAY**: A vector of \( n \) elements with soil clay content (particle diameters \( <= 2 \times 10^{-6} \) m), in percent \([0, 100]\).
- **SAND**: A vector of \( n \) elements with soil sand content (particle diameters \(< 2 \) mm and \( > 50 \times 10^{-6} \) m), in percent \([0, 100]\).
- **BD**: A vector of \( n \) elements with soil bulk density (g/cm³).
- **OC**: A vector of \( n \) elements with soil organic carbon content, in percent \([0, 100]\).

Details
Pedotransfer function returns the van Genuchten-Mualem model parameters given CLAY, SAND, BD, and OC. The correction of the original paper presented by Weynants et al (2009), were made by Weihermueller et al., (2017), which is implemented.

Value
List with the van Genuchten-Mualem parameters, each as a vector of \( n \) elements:
- **thr**: Residual water content (-), always equal to zero
- **ths**: Saturated water content (-)
- **alp1**: Shape parameter \((\text{cm}^{-1})\)
- **n1**: Shape parameter (-)
- **K0**: Hydraulic conductivity at 0 potential \((\text{cm/day})\)
- **tau**: Shape parameter (-)
Note

The PTF is not suitable for predicting the hydraulic conductivity curve at pressured heads > -6 cm. (Weynant et al., 2009)

Author(s)

Melanie Weynants <mweynants@gmail.com> Tobias K.D. Weber <tobias.weber@uni-hohenheim.de>

References


Examples

ptf.cW(CLAY = .4, SAND = .4, BD = 1.6, OC = .5)

---

ptf.vG2FM

Parameter Transfer Function for Weber et al. (2019) model

Description

Predicts Weber et al. (2019) model parameters in the van Genuchten-Mualem variant 0I110FM, from previously obtained van Genuchten-Mualem parameters for the constrained van Genuchten-Mualem model.

Usage

ptf.vG2FM(x)

Arguments

x

A vector of 6 van Genuchten-Model parameters.

Details

Pedotransfer function returns the van Genuchten - Mualem model 0I110 parameters in the Brunswick-Model variant 0I110FM, based on previously determined van Genuchten-Mualem parameters. The transfer function is based on an ordinary linear regression between the i-th 0I110 and 0I110FM. The parameters were based on model fits to a dataset of ~200 samples with retention and conductivity measurements.
Value

vector of van Genuchten-Mualem model parameters, the order of which is sensitive.
Note

The parameter transfer function was derived by ordinary linear regression, by regressing the estimated the Weber et al. (2018) framework model parameters Genuchten Mualem variant from measured soil water retention and hydraulic conductivity data against the constrained van Genuchten model parameters. The regression of \( \alpha \) and \( K_s \), and \( (n-1) \) was done in the \( \log(10) \)-transformed space, and \( K_{ncs} \) is predicted as the fraction \( w \) of \( K_s \).

Author(s)

Efstathios Diamantopoulos <ed@plen.ku.dk> Tobias K.D. Weber <tobias.weber@uni-hohenheim.de>

References


Examples

```r
p = c(0.08, 0.42, 0.01, 1.5, 100, 0.5)
ptf.vG2FM(p)
```

---

**Description**

Contains the objective functions to calculate (weighted) sum of squared residuals or likelihoods. The assumption made is that the residuals are identically, independently and normally distributed. The normal distribution of the model residuals is standardly given with mean = 0, and variance = 1. if weighting is not considered. There are three methods to consider weights: a) fixed scalar values for each data type, b) a vector of weights for each data type. The vector has to have the same length as the vector of the data type. c) It is possible to estimate the

**Usage**

```r
resFun(p, shpmodel = "01110", retdata = NULL, condata = NULL, pretrans = NULL, weight = NULL, method = "rss", trim.query = FALSE, ivap.query = NULL, hclip.query = FALSE, parL = NULL)
```
**Arguments**

- **p**
  Vector of model parameters handed used to calculate the soil hydraulic property model values in `shypFun`. Depends on `shpmodel` and the pressure head values specified in `retdata` and `condata`.

- **shpmodel**
  Character identifying the soil hydraulic property model. See `shypFun`.

- **retdata**
  A dataframe or matrix with 2 columns. The first with pressure head values in [cm] and the second with volumetric water contents in [cm cm-3].

- **condata**
  A dataframe or matrix with 2 columns. The first with pressure head values in [cm] and the second with hydraulic conductivity values in log10[cm d-1].

- **pretrans**
  A vector to back transform the parameters before the soil hydraulic property function values calculated.

- **weight**

"rss" default for the optimisation algorithm DEoptim. `resFun` returns scalar sum of squared (weighted) residuals. "res" `resFun` returns a list with vectors of weighted residuals. Required for post hoc analyses.

- **method**
  **trim.query** default FALSE. If a trimodal soil hydraulic property function is used, this has to be specified by setting the argument to (TRUE) which ensures the sum of modal weights == 1.

- **ivap.query**
  Default is FALSE, and noivap method is specified. See `KvapFun`.

- **hclip.query**
  Implemented purely for future compatibility. Currently no use.

- **parL**
  Defaults to NULL, only inserted for compatibility with modMCMC used in `shypEstFun`. modMCMC, only handled parameters which are estimated, other model parameters need to be passed through parL. See Details of `shypEstFun`.

**Details**

Model errors may be specified or estimated as nuisance parameters weighting the data classes. In case the model error !=1, the output statistics are weighted accordingly.

- **user**
  User defined weights.

- **none**
  No weights are considered, i.e. no measurement error assumed.

- **range**
  Rescaling (normalization of observations to the intervall [0,1].

- **fixl**
  Fixed scalar weight for THETA is 1/0.05^2 and weight for log10K is 1.

- **estl**
  Two scalar model weights 1/sigma_i^2 are treated as free parameters to be estimated by inversion, one for THETA and

**Value**

Returns scalar of sum of squared (weighted) residuals or vector of weighted residuals, as specified by

- **rss**
  Scalar sum of squared (weighted) residuals.
res vector of weighted residuals
loglik log-likelihood value

Note
Calculates the objective function value as the sum of squared weighted model errors.

Author(s)
Tobias KD Weber

References

Examples

# load data
data("shpdata1")

# observations
retdata <- shpdata1$LFH1$wrc[!is.na(shpdata1$LFH1$wrc[,1]),]
conda <- shpdata1$LFH1$hcc

# 7 - resFun ------------------------------------------
# soil hydraulic property model parameters, van Genuchten-Mualem
p <- c("thr" = 0.16, "ths" = 0.46, "alf" = 0.03, "n1" = 1.42, "Ks" = 26, "tau" = .5)

# calculate weighted residuals
wres <- resFun(p, retdata = retdata, condata = condata, pretrans = NULL, weight = list("wth" = 0.0025, "wKh" = 1), method = "res", trim = FALSE)

## residuals of the soil water retention curve [-]
theta.wres <- wres[1:dim(retdata)[1]]

## residuals of the log10 hydraulic conductivity curve [cm/d]
log10K.wres <- wres[(dim(retdata)[1]+1) : length(wres)]

shpdata1 Measured soil hydraulic property data

Description
Data from evaporation experiments using the UMS HYPROP device on two soils with different textures Data is reported in [cm3 cm-3]
**Usage**

data(shpdata1)

**Format**

An object of class `list`.

**Source**

none

**References**


**Examples**

data(shpdata1)
ticksatmin <- -2
tcllen <- 0.4
ticksat <- seq(ticksatmin,5,1)
pow <- ticksatmin:5

TS.wrc <- shpdata1$TS1$wrc; TS.hcc <- shpdata1$LFH1$hwc
TS.hcc <- shpdata1$TS1$hcc; LFH.hcc <- shpdata1$LFH1$hcc

# PLOT THE MEASURED WATER RETENTION CURVE
plot(TS.wrc[,1:2] , ylim = c(.1,.50), xlim = c(0,4), ylab = "", xlab = "",
col = "darkgrey", axes=FALSE, main = "Measured Water Retention Curve")
points(TS.wrc[,1:2],pch = 4,col = "darkblue")
legend("topright", c("TS1", "LFH1"), pch = c(1,4), bty = "n", cex=1.2,
col = c("darkgrey","darkblue"))
axis(1, at = pow, labels = parse(text = paste("10^",(pow), sep = "")),tcl = tcllen)
axis(2, tcl = tcllen)
axis(4, labels = NA, tcl = tcllen)
axis(3, labels = NA, tcl = tcllen)
mtext("pressure head [h] [cm]", 1, cex = 1.2, line = 2.8)
mtext("vol. water content [ - ]", 2, cex = 1.2, line = 2.8)
box()

# PLOT THE MEASURED HYDRAULIC CONDUCTIVITY CURVE

plot(TS.hcc, ylim = c(-8.2), xlim = c(0,4), ylab = "", xlab = "",
col = "darkgrey",
axes = FALSE, main = "Measured Hydraulic Conductivity Curve")
points(TS.hcc, pch = 4, col = "darkblue")
legend("topright", c("TS1", "LFH1"), pch = c(1,4), bty = "n", cex = 1.2,
col = c("darkgrey","darkblue"))
axis(1, at = pow, labels = parse(text = paste("10^",(pow), sep = "")), tcl = tcllen)
Description

Estimates model parameters of implemented soil hydraulic property functions. Various additional options exist.

This function sets up the parameter estimation, given a set of arguments, and enables minimisation of (weighted) sum of squared residuals, assuming independent and identically distributed model residuals.

More information on the options is given in the Details

Usage

shypEstFun(shpmodel = "01110", parl, retdata, condata,
  ivap = NULL, hclip = FALSE,
  weightmethod = "none", LikModel = "rss",
  ALG = "DE", set.itermax = 200, ALGoptions, lhs.query = FALSE)

Arguments

Character specifying the soil hydraulic property model. Currently valid models as documented in shypFun and are:

01110 constrained unimodal van Genuchten-Mualem.
01210 constrained bimodal van Genuchten-Mualem.
01310 constrained trimodal van Genuchten-Mualem.
02110 unimodel Kosugi 2 parametric-Mualem model (Kosugi, 1996)
03110 unimodel Fredlund-Xing-Mualem model, with the contraint of m = 1-1/n (Fredlund D.G., and A. Xing, 1994)

shpmodel
parl  list of 4 vectors with named vectors, the order in the list is sensitive.
p  vector with length 1 of model specific initial parameters, has to coincide with the chosen soil hydraulic property model
pse  vector with length 1 identifying which parameters are to be estimated
plo  vector of lower bounds (non-transformed parameter boundaries)
pup  vector of upper bounds (non-transformed parameters boundaries)
Alternatively, a list of vectors can be provided specifying the user given model weights ($1/\sigma^2$). Either as skalar for each data class, or a vector with the same length as the number of data points given for each of the measurements in the respective data class.

retdata A dataframe or matrix with 2 columns. The first with $\log_{10}$ values of pressure head values in [cm] and the second with volumetric water contents in [cm cm$^{-3}$].

condata A dataframe or matrix with 2 columns. The first with $\log_{10}$ values of pressure head values in [cm] and the second with hydraulic conductivity values $\log_{10}$[cm d$^{-1}$].

ivap Specification if isothermal vapour conductivity after Saito et al. (2006) should be accounted for, defaults to NULL and no isothermal vapour conductivity is considered.

hclip Specification if the hydraulic conductivity model should be 'clipped', i.e. constrained to a maxium pore diamater as introduced by Iden et al. (2015), defaults to FALSE This has been implemented for future development reasons and is not yet functional.

weightmethod Specification of weight method. The implemented methods are

none no weights are considered, i.e. no measurement error assumed
range normalization of observations to the intervall [0,1]
fix1 fixed scalar weight for THETA is 0.05$^2$ and weight for $\log_{10}$K is 1
est1 Two scalar model weights ($1/\sigma^2$) are treated as free parameters to be estimated by inversion, one for THETA and

Alternatively, a list of vectors can be provided specifying the user given model weights ($1/\sigma^2$). Either as skalar for each data class, or a vector with the same length as the number of data points given for each of the measurements in the respective data class. The length of the list has to coincide with the data groups.

LikModel Specification of inverse modelling type. Has to be specified but implemented for future compatability)

rss Default for the optimisation algorithm DEoptim. resFun returns scalar sum of squared (weighted) residuals

$-2\log\text{lik}$ Specified if ALG == $-2+\log-\text{likelihood value, which is minimised assuming Gaussian and i.i.d (weighted) residuals

ALG Select global optimisation algorithm or a Markov chain Monte Carlos (MCMC) sampler.

DE Default for the optimisation algorithm DEoptim. resFun returns a scalar sum of squared (weighted) residuals if LikModel == NULL

modMCMC Default for the DRAM (Delayed Rejection Adaption Metropolis) algrothim implemented in modMCMC of the FME package.

set.itermax Integer specifying the maximum number of iterations.

ALGoptions A list with named entries setting the algorithm options. Each list element name is required to be identical with the names as documented in the respective algo-
rtihm help DEoptim.control and modMCMC.

set.itermax overrides the maximum iterations argument.

lhsNquery default FALSE, TRUE will produce a Latin Hypercube Sample for the initial population when using DEoptim.

Details

Several in-built methods for weighting the (multi-) objective function residuals are available, they may be specified, or estimated as nuisance parameters for the two data groups. More details see weightfun. Weights are the inverse of the squared standard deviation of the residuals (variance).

Generally, soil hydraulic property model parameters are estimated as transformed parameters: log10 for alpha_i, Ks, and log10 for n_i-1, Kc, Knc

For model codes in ivap please refer to KvapFun

Parallel computing for package DEoptim is not supported. And the optional arguments in modMCMC are not supported.

Value

list with the return of the optimisation algrorithm or MCMC sampler and all settings.

settings a list with output of the optimisation and summary of settings:

weigh the list with weights for the retention and conductivity data.
parL the list of initial and selected model parameters, and upper and lower bounds.
transL list of parameter transformation rules used
shpmodel the used soil hydraulic property model
ivap isothermal vapour conductivity model
hclip for future compatability
LikModel the adopted method to calculate the objective function value
data a list of 2 objects with a) retention data and b) conductivity data used for the parameter estimation.

out result of algorithm function DEoptim or modMCMC

Note

The function is currently set up that parameters can only be estimated if both retention and conductivity data are given

Author(s)

Tobias KD Weber

References


See Also

For more details on `shpmodel` please look at `shypFun`

Examples

data("shpdata")
retdata <- shpdata$TS1$wrc
codata <- shpdata$TS1$hcc
codata <- codata[!is.na(codata[,1]),]
# set set.itermax higher.
weightmethod = "range"
ivap = NULL
set.itermax = 1
LikModel = "rss" # ALTERNATIVE OPTION: LikModel = "-2logLik"
ALG = "DE" # ALTERNATIVE OPTION: ALG = "modMCMC"

parL <- list("p" = c("thr" = 0.05, "ths" = 0.45, "alf1" = 0.01, "n" = 2, "Ks" = 100, "tau" = 0.5),
"psel" = c(1, 1, 1, 1, 1),
"plo" = c(0.001, 0.2, 0.001, 1.1, 1, -2),
"pup" = c(0.3, 0.8, .1, 11, 1e4, 10))

out <- shypEstFun(shpmodel = "01110",
parL = parL,
retdata = retdata, condata = codata,
ivap = ivap,
hclip = FALSE,
weightmethod = weightmethod,
LikModel = LikModel,
ALG = ALG,
set.itermax = set.itermax,
lhs.query = FALSE)

data("shpdata")
retdata <- ret <- shpdata$TS1$wrc
codata <- con <- shpdata$TS1$hcc
conda <- condata[!is.na(conda[,1])]

---

# 1 SET VARIABLES -----------------------------------------------

# VARIABLES FOR PLOTTING

(pF <- seq(-3, 6.8, length = 201))
h <- 10^pF
ticksatmin <- -2
tcllen <- 0.4
ticksat <- seq(ticksatmin, 5, 1)
pow <- ticksatmin:5

# VARIABLES FOR THE FITTING ALGORITHM

weightmethod = "range"
ivap = NULL
set.itermax = 3e1
LikModel = "rss" # ALTERNATIVE OPTION: LikModel = "-2logLik"
ALG = "DE" # ALTERNATIVE OPTION: ALG = "modMCMC"
shpmodel.v <- c("01110", "01110FM")

plotNquery = FALSE
no.shps <- length(shpmodel.v)

# initialising lists

out.L <- vector("list", no.shps)
gof.L <- vector("list", no.shps)

# Run comparison

for (i in 1:2) {
  shpmodel = shpmodel.v[i]
  switch(shpmodel,
    "01110" = {
      # van Genuchten-Mualem Model parameters
      parL<-list("p"=c("thr"=0.05,"ths"=0.45,"alf1"=0.01,"n"=2,"Ks"=100,"tau"=.5), 
                  "psel" = c(1, 1, 1, 1, 1, 1),
                  "plo"=c(0.001 , 0.2 , 0.001 , 1.1, 1, -2),
                  "pup"= c(0.3 , 0.8 , 11, 1e4, 10)
    },
    "01110FM" = {
      # van Genuchten-Mualem Model parameters + BRUNSWICK MODEL (WEBER ET AL. 2019, WRR, in rev)
      parL<-list("p"=c("thr"=0.05,"ths"=0.45,"alf1"=0.01,"n"=2,"Ksc"=100, 
                      "tau"=.5,"Kscn"=1e-4,"a"=1.5,"h0"=6.8),
                      "psel" = c(1, 1, 1, 1, 1, 1, 0, 0),
                      "plo"=c(0.001 , 0.1 , 0.001 , 1.1, 1,0,1e-6 , 1, 6.5),
                      "pup"= c(0.35 , 0.7 , 11, 1e4,10 ,1e0, 2, 6.9)
    })
  )
}

stop("Enter a meaningful shpmodel")
)

out <- shypEstFun(shpmodel = shpmodel,
parL = parL,
retdata = retdata, condata = condata,
ivap = ivap,
hclip = FALSE,
weightmethod = weightmethod,
LikModel = LikModel,
ALG = ALG,
set.itermax = set.itermax
, lhs.query = FALSE)

out$model <- shpmodel.v[i]
out.L[[i]] <- out

# Calculate the soil hydraulic properties for the plot
if(ALG == "DE"){
  p <- out$out$optim$phattrans
}

if(ALG == "modMCMC"){
  p <- out$out$bestpartrans
}

if(weightmethod == "est1"){
  np <- length(p)
  p <- p[-c(np-1, np)]
  if(ALG == "modMCMC"){
    parL$p[which(parL$p$sell==1)] <- p
    p <- parL$p
  }
}

if(plot.query==TRUE){
  shyp.L<-shypFun(p,h,shpmodel=shpmodel.v[i],ivap.query=ivap)
  if(shpmodel == c("01110")){
    wrc<-shyp.L$theta
    hcc<-log10(shyp.L$Kh)
  }
}

# PLOT THE WATER RETENTION CURVE
par(mfrow=c(1,2), tcl=tcllen)
plot(retdata, ylim=c(.0, .50), xlim=c(0, 6.8), ylab="", xlab="",
col="darkgrey", axes=FALSE, main="Water Retention Curve", cex=2)
lines(log10(abs(h)), wrc, col="darkblue", lwd=2)
legend("topright", c("observed", "simulated"),pch=c(1,NA),
  lty=c(NA,1), lwd=2, bty="n", cex=1.3, col=c("darkgrey","darkblue"))
axis(1, at=pow, labels=parse(text="paste("10",(pow),sep="", tcl=tcllen)"
# PLOT THE MEASURED HYDRAULIC CONDUCTIVITY CURVE

```r
plot(conda, ylim=c(-8, 2), xlim=c(0, 6.8), ylab="", xlab="", col="darkgrey",
     axes=FALSE, main="HydraulicConductivityCurve", cex=2)
lines(log10(abs(h)), hcc, col="darkblue", lwd=2)
legend("topright", c("observed", "simulated"), pch=c(1, NA),
       lty=c(NA, 1), lwd=2, lty="n", cex=1.3, col=c("darkgrey", "darkblue"))
axis(1, at=pow, labels=parse(text=paste("10^",(pow), sep="")), tcl=tclllen)
axis(2)
axis(4, labels=NA)
axis(3, labels=NA)
```

```r
# PLOT THE HYDRAULIC CONDUCTIVITY CURVE

```
shypFun

Wrapper Function for all Supported Soil Hydraulic Property Models.

Description

This function allows to select a soil hydraulic property model.

Usage

shypFun(p, h, shpmodel = "01110", ivap.query = NULL)

Arguments

p
Vector of the 6 van Genuchten-Mualem model parameters, order is sensitive. cf
respective model documentation

h
Pressure heads [cm] for which the corresponding retention and conductivity val-
ues are calculated

shpmodel
character

01110 unimodel van Genuchten-Mualem model, with the contraint of m = 1-1/n (van Genuchten, 1980)
shypFun

01210 bimodel van Genuchten-Mualem model, with the constraint of \( m_i = 1 - 1/n_i \) (Durner, 1994)
01310 trimodal van Genuchten-Mualem model, with the constraint of \( m_i = 1 - 1/n_i \) (Durner, 1994)
02110 unimodel Kosugi 2 parametric-Mualem model (Kosugi, 1996)
03110 unimodel Fredlund-Xing-Mualem model, with the constraint of \( m = 1 - 1/n \) (Fredlund D.G., and A. Xing, 1994)

NULL no isothermal vapour conductivity will be calculated with Kvap
Model type for isothermal vapour conductivity, see Details of function KvapFun for model codes

ivap.query

Details

If the shpmodel code is supplemented by FM, e.g. shpmodel = "01110FM", the Framework-Model (Brunswick-Model) will be activated. Ensuring a water content of 0 at oven dryness (at a pressure head of \( h_0 = 10^6.8 \) cm), continuous differentiability of the retention curve, accounting for capillary and non-capillary water storage and conductivity by introducing only one additional model parameter, and the possibility to model a the often observed change in slope in the conductivity model in the medium pressure head range.

Value

returns a list with calculations at specified \( h \):

theta calculated volumetric moisture content
Se calculated saturation
Scap effective saturation (of the capillary part if FM is specified)
cap specific water capacity function (of the capillary part if FM is specified)
psd pore size distribution (of the capillary part if FM is specified)
Kh total hydraulic conductivity

if FM specified, additionally:

thetacap calculated volumetric moisture content of the capillary part
thetanc calculated volumetric moisture content of the non-capillary part
Snc effective saturation of the non-capillary part
Kcap hydraulic conductivity of the capillary
Knc hydraulic conductivity of the non-capillary
Kvap isothermal vapour conductivity
Krcap relative hydraulic conductivity of the capillary
Krnc relative hydraulic conductivity of the non-capillary
Note

the function is used to assign a new function variable with a function which calculates the soil hydraulic properties according to specified shpmodel and model specified by ivap.query

Author(s)

Tobias KD Weber

References


See Also

PC-Progress RETC

Examples

# load measured data
data("shpdata")
retdata <- shpdata$sFH$wrc[][is.na(shpdata$sFH$wrc[,1]),]
condata <- shpdata$sFH$hcc

# assign auxiliary variables
\pf <- seq(-3, 6.8, length = 501)
\h <- 10^{\pf}

# assign list of parameters for the van Genuchten-Mualem model
parL <- list("p" = c("thr" = 0.05, "ths" = 0.45, "alf1" = 0.01, "n" = 2, "Ks" = 100, "tau" = .5),
             "pse1" = c(1, 1, 0, 1, 1, 1),
             "pio" = c(0.001, 0.2, 0.001, 1.1, 1, -2),
             "pup" = c(0.3, 0.95, 1, 10, 1e4, 10))
# calculate soil hydraulic property function values

`shypFun(parL$p, h, shpmodel = "01110", ivap.query = NULL)`

`wrc <- shyp.L$theta`

`hcc <- log10(shyp.L$Kh)`

# # PLOT THE MEASURED WATER RETENTION CURVE

`ticksatmin <- -2`

`tcllen <- 0.4`

`ticksat <- seq(ticksatmin, 5, 1)`

`pow <- ticksatmin:6`

`par(mfrow = c(1,2), tcl = tcllen)`

`plot(retdata, ylim = c(0, .5), xlim = c(0, 6.8), ylab = "", xlab = ", col = "darkgrey", axes = FALSE, main = "Water Retention Curve", cex = 2)`

`lines(log10(h), wrc, col = "darkblue", lwd = 2)`

`legend("topright", c("observed", "simulated"),pch = c(1,NA), lty = c(NA,1), lwd = 2, bty = "n", cex = 1.3,col = c("darkgrey", "darkblue"))`

`axis(1, at = pow, labels = parse(text = paste("10^",(pow), sep = "")), tcl = tcllen)`

`axis(2, tcl = tcllen)`

`axis(4, labels = NA)`

`axis(3, labels = NA)`

`mtext("pressure head |h| [cm]", 1, cex = 1.2, line = 2.8)`

`mtext("vol. water content [ - ]", 2, cex = 1.2, line = 2.8)`

`box()`

# # PLOT THE MEASURED HYDRAULIC CONDUCTIVITY CURVE

`plot(entrada, ylim = c(-8,2), xlim = c(0, 6.8), ylab = "", xlab = "", col = "darkgrey",`  

`axes = FALSE, main = "Hydraulic Conductivity Curve", cex = 2)`

`lines(log10(h), hcc, col = "darkblue", lwd = 2)`

`legend("topright", c("observed", "simulated"),pch = c(1,NA), lty = c(NA,1), lwd = 2, bty = "n", cex = 1.3,col = c("darkgrey", "darkblue"))`

`axis(1, at = pow, labels = parse(text = paste("10^",(pow), sep = "")), tcl = tcllen)`

`axis(2)`

`axis(4, labels = NA)`

`axis(3, labels = NA)`

`mtext("log10 K [cm/d]", 2, cex = 1.2, line = 2.8)`

`mtext("pressure head |h| [cm]",1 , cex = 1.2, line = 2.8)`

`box()`

`par(mfrow = c(1,1))`

# # HOW TO WRITE A MATER.IN FOR HYDRUS-1D

`mater_out <- cbind(shyp.L[['theta']], h, shyp.L[['Kh']], abs(shyp.L[['cap']])))`

`materWriteFun <- function(mater_out.L, path = getwd(), sample) {

  # Function to write a Mater.in

  # ARGUMENTS

}
shypFun.01110

van Genuchten-Mualem Function

Description
Calculates the soil hydraulic property function values based on given pressure heads

Usage
shypFun.01110(p, h)

Arguments
p vector of the 6 van Genuchten-Mualem model parameters, order is sensitive and has to be given as:

thr residual water content [cm cm-3]
ths saturated water content [cm cm-3]
alfl van Genuchten alpha [cm-3]
n1 van Genuchten n [-]
Ks saturated conductivity [cm d-1]
tau exponent of Se in the capillary conductivity model, sometimes denoted in the literature as l [-]

h pressure heads [cm] for which the corresponding retention and conductivity values are calculated

Details
The function solves analytically the spec. water capacity function and integral to the capillary bundle model
Value
returns a list with calculations at specified h:
theta calculated volumetric moisture content
Se calculated saturation
cap specific water capacity function
psd pore size distribution
Kh Hydraulic conductivity values

Author(s)
Tobias KD Weber

References

Examples
p <- c(0.1, 0.4, 0.01, 2, 100, .5)
h <- 10*seq(-2, 6.8, length = 197)
shyp.L <- shypFun.01210(p, h)

Description
Calculates the soil hydraulic property function values based on given pressure heads

Usage
shypFun.01210(p, h)

Arguments
p vector of the 9 bimodal van Genuchten-Mualem model parameters, order is sensitive and has to be given as:
thr residual water water content [cm cm-3]
ths saturated water water content [cm cm-3]
alfl van Genuchten alpha [cm-3]
n1 van Genuchten n [-]
w1 fraction of the first modality [-]. w2 is internally computed as w2 = 1-w1
alfl2 van Genuchten alpha of the second modality [cm-3]
\( n_2 \)  \( \text{van Genuchten n of the second modality [-]} \)
\( K_s \)  \( \text{saturated conductivity [cm d^{-1}]} \)
\( \tau_{\text{au}} \)  \( \text{exponent of } S_e \text{ in the capillary conductivity model, sometimes denoted in the literature as } l [-] \)

\( h \)  \( \text{pressure heads [cm] for which the corresponding retention and conductivity values are calculated} \)

**Details**

The function solves analytically the specific water capacity function and integral to the capillary bundle model

**Value**

returns a list with calculations at specified \( h \):

- \( \theta \)  \( \text{calculated volumetric moisture content} \)
- \( S_e \)  \( \text{calculated saturation} \)
- \( \text{cap} \)  \( \text{specific water capacity function} \)
- \( \text{psd} \)  \( \text{pore size distribution} \)
- \( K_h \)  \( \text{Hydraulic conductivity values} \)

**Author(s)**

Tobias KD Weber

**References**


**Examples**

```r
p <- c("thr" = 0.1, "ths" = 0.4, "alf1" = 0.5, "n1" = 3, "w1" = .6, "alf2" = 0.01, "n2" = 1.6, "Ks" = 100, "tau" = .5)

h <- 10*seq(-2, 6.8, length = 197)

shyp.L <- shypFun.01210(p, h)
```
**shypFun.01310**  
*Durner Model (trimodal van Genuchten Mualem) Function*

**Description**
Calculates the soil hydraulic property function values based on given pressure heads.

**Usage**

```r
shypFun.01310(p, h)
```

**Arguments**
- `p`: vector of the 9 bimodal van Genuchten-Mualem model parameters, order is sensitive and has to be given as:
  - `thr`: residual water water content [cm cm\(^{-3}\)]
  - `ths`: saturated water water content [cm cm\(^{-3}\)]
  - `alfQ`: van Genuchten alpha [cm\(^{-3}\)]
  - `nQ`: van Genuchten n [-]
  - `wQ`: fraction of the first modality [-], `wR` is internally computed as `wR = QMwQ` in `resfun` ensures `wS > 0`
  - `alfR`: van Genuchten alpha of the second modality [cm\(^{-3}\)]
  - `nR`: van Genuchten n of the second modality [-]
  - `wR`: fraction of the second modality [-], `wS` is internally computed as `wS = 1 - w1 - w2`, in `resfun` ensures `w3 >= 0`
  - `alfS`: van Genuchten alpha of the third modality [cm\(^{-3}\)]
  - `nS`: van Genuchten n of the third modality [-]
  - `ks`: saturated conductivity [cm d\(^{-1}\)]
  - `tau`: exponent of Se in the capillary conductivity model, sometimes denoted in the literature as \( \lambda \) [-]

- `h`: pressure heads [cm] for which the corresponding retention and conductivity values are calculated

**Details**
The function solves analytically the specific water capacity function and the integral to the capillary bundle model.

For applications of the trimodal model, eg.: Weber et al. (2017a, 2017b).

**Value**
returns a list with calculations at specified `h`:
- `theta`: calculated volumetric moisture content
- `Se`: calculated saturation
- `cap`: specific water capacity function
**Kosugi-Mualem Model (2 Parameter Model)**

**Description**

Calculates the soil hydraulic property function values based on given pressure heads

**Usage**

```r
shypFun.02110(p, h)
```

**Arguments**

- `p`: vector of the 6 Kosugi-Mualem model parameters, order is sensitive and has to be given as:

```r
p <- c("thr" = 0.1, "ths" = 0.4, alf1 = .5, "n1" = 3,
"w1" = .5, "alf2" = 0.01, "n2" = 2,
"w2" = .3, "alf3" = 0.01, "n3" = 1.6,
"Ks" = 100, "tau" = .5)
```

**Examples**

```r
h <- 10*seq(-2, 6.8, length = 197)
shyp.L <- shypFun.01310(p, h)
```
thr residual water water content [cm cm^{-3}]
ths saturated water water content [cm cm^{-3}]
hm air entry pressure head [cm]
sigma width of pore size distribution [ - ]
Ks saturated conductivity [cm d^{-1}]
tau exponent of Se in the capillary conductivity model, sometimes denoted in the literature as 1 [-]

h pressure heads [cm] for which the corresponding retention and conductivity values are calculated

Details
The function solves analytically the spec. water capacity function and integral to the capillary bundle model

Value
returns a list with calculations at specified h:

theta calculated volumetric moisture content
Se calculated saturation
cap specific water capacity function
psd pore size distribution
Kh Hydraulic conductivity values

Author(s)
Tobias KD Weber

References

Examples

\[(p, h) <- \text{c("thr" = 0.1, "ths" = 0.4, "hm" = 100, "sigma" = 2, "Ks" = 100, "tau" = 0.5), h <- 10*\text{seq}(0, 10, length = 197), shyp.L <- \text{shypFun.02110}(p, h)}\]
Description

Calculates the soil hydraulic property function values based on given pressure heads. The function calculates the base function of Fredlund and Xing.

Usage

\[ \text{shypFun.03110}(p, h) \]

Arguments

\( p \) vector of the 6 Fredlund-Xing model parameters, order is sensitive and has to be given as:

- \( \text{thr} \) residual water water content [cm cm\(^{-3}\)]
- \( \text{ths} \) saturated water water content [cm cm\(^{-3}\)]
- \( \text{alf} \) inverse of the air entry pressure head [cm]
- \( n_1 \) width of pore size distribution [ - ]
- \( K_s \) saturated conductivity [cm d\(^{-1}\)]
- \( \tau \) exponent of \( S_e \) in the capillary conductivity model, sometimes denoted in the literature as \( l \) [ - ]

\( h \) pressure heads [cm] for which the corresponding retention and conductivity values are calculated

Details

The function numerically solves the specific water capacity function and integral to Mualem’s conductivity model.

Value

returns a list with calculations at specified \( h \):

- \( \text{theta} \) calculated volumetric moisture content
- \( \text{Se} \) calculated saturation
- \( \text{cap} \) specific water capacity function
- \( \text{psd} \) pore size distribution

Author(s)

Tobias KD Weber
References


Examples

```r
p <- c(0.1, 0.4, 0.01, 2, 100, 0.5)
h <- 10*seq(-2, 6.8, length = 197)
shyp.L <- shypFun.03110(p, h)
```

sncFun

Non-capillary Saturation Function

Description

The general purpose method to calculate the effective non-capillary saturation is directly obtained from any arbitrary expression for the rescaled capillary saturation function as described by Weber et al. (2019). Examples of capillary saturation functions are the well known van Genuchten (1980), Fredlund and Xing (1993), and Kosugi (1996) functions.

Usage

```r
sncFun(h, scap)
```

Arguments

- `h`: A vector of \( n \) pressure head values for which `scap` was calculated
- `scap`: vector of \( n \) monotonically decreasing capillary saturation function values calculated by `shypFun`, rescaled between 0 and 1.

Details

More details in Weber et al. (2019)

Value

- `snc`: A vector of \( n \) element with calculated volumetric moisture content

Note

The function requires a numerical solution to the integral of Eq. 6 in Weber et al. (2018) and; therefore, it is advisable to use a sufficient amount of pressure head data values to minimise the numerical error.

Author(s)

Tobias KD Weber
References


Examples

```r
# set variables
p <- c(0.1, 0.4, 0.01, 2, 100, .5)
h <- 10^seq(-2, 6.8, length = 197)

# Calculate the capillary and non-capillary saturation function.
Se <- shypFun(p, h, shpmodel = "01110")$Se
Snc <- sncFun(Se)
```

---

transBoundFun

*Creates Parameter Transformation and Backtransformation Rules for the Estimation Procedure*

Description

This function is intended for the function shypEstFun so that lists with set rules for the transformation and back-transformation of the soil hydraulic model parameters are enabled. In general, the following rules apply log\(\theta\) transformation for the model parameters \(\alpha, n_i=1, K_s, K_{sc}, K_{snc}\).

Usage

```r
transBoundFun(parL, shpmodel, weightmethod)
```

Arguments

- **parL**: a list with 4 numeric vectors specifying:
  - **p**: Vector of model parameters, has to coincide with the chosen soil hydraulic property model. If `weightmethod == est` then two additional nuisance parameters, need to be specified and concatenated to the vector of soil hydraulic property model parameters, a first, for THETA and a second for log10K)
  - **psel**: vector identifying which parameters are to be estimated
  - **plo**: vector of lower bounds (non-transformed parameter boundaries)
  - **pup**: vector of upper bounds (non-transformed parameters boundaries)
  - **shpmodel**: A string specifying the selected shp model.
weightmethod  A string specifying the selected weighting method, if weightmethod == "est1" is TRUE, then parl is modified to account for nuisance parameters).

Details

The function is meant for internal use in shypEstFun.

Value

Returns a list of two lists. One of the sub-lists is parl but with transformed parameters, and the second, transl with model specific transformation and back-transformation rules.

Note

This function is meant for internal use but can be used regardless.

Author(s)

Tobias KD Weber

Examples

# List of model parameters
parL <- list("p" = c("thr"= 0.05, "ths" = 0.45, "alf1" = 0.01, "n" = 2, "Ks" = 100, "tau" = .5),  
"psel" = c(1, 1, 0, 1, 1, 1),  
"plo" = c(0.001, 0.2, 0.001, 1.1, 1, -2),  
"pup" = c(0.3, 0.95, 1, 10, 1e4, 10))

define transformation and back-transformation of parameter vectors
for(k in c("p", "plo", "pup")){
  for (j in c("none")){
    parl.trans <- transBoundFun(parL, shpmodel = "01110", weightmethod = j)
    p_trans <- transFun(parL[[k]], parl.trans$transL$ptrans)
    p_retrans <- transFun(p_trans, parl.trans$transL$pretrans)
    stopifnot(sum(p_retrans != parL[[k]])==0)
  }
}

transFun

Parameter Transformation and Backtransformation

Description

Enables the transformation and backtransformation of parameters. This is widely considered advantageous during parameter estimation as the parameter space in the transformed is well-behaved, e.g. with normally distributed posteriors.
transFun

Usage

transFun(par.vec, trans.L)

Arguments

par.vec Vector of n model parameters
trans.L list of n transformation/backtransformation operators, transformation and backtransformation rules have to be antonyms and position in vector has to coincide with that in par.vec

Details

Transformation rules are:

\[ \log_{10}\alpha_i, \log_{10}n_i - 1, \log_{10}K_s, \log_{10}\omega, \log_{10}K_{sc}, \text{and} \log_{10}K_{snc} \]

Value

p.transformed Returns transformed parameters as specified by trans.L

Note

The function is used to transform the parameter space and enabling optimisation or MCMC sampling to be more efficient.

Author(s)

Tobias KD Weber

Examples

# van Genuchten-Mualem Model parameters
parL <- list("p" = c("thr" = 0.05, "ths" = 0.45, "alf1" = 0.01, "n" = 2, "Ks" = 100, "tau" = .5),
"psel" = c(1, 1, 0, 1, 1, 1),
"plo" = c(0.001, 0.2, 0.001, 1.1, 1, -2),
"pup" = c(0.3, 0.95, 1, 10, 1e4, 10)
)
# Two lists, one with function to transform, the other to back-transform model parameters
ptransfit <- c(function(x)x, function(x)x, log10, function(x)log10(x-1), log10, function(x)x)
pretransfit <- c(function(x)x, function(x)x, function(x)x, function(x)x, function(x)x)
# Transform
p_trans <- transFun(parL$p, ptransfit)
weightFun

**Specification of Weights for the Data Groups Retention Data and Conductivity Data.**

**Description**

Weights can be fixed to suggested standards, fixed by the user, or estimated as additional nuisance parameters.

**Usage**

weightFun(weightmethod = "fix1", retdata, condata, parl = NA)

**Arguments**

- **weightmethod** character specifying the method of selecting model weights
- **retdata** a dataframe or matrix with 2 columns. The first with pressure head values in [cm] and the second with volumetric water contents in [cm cm-3].
- **condata** a dataframe or matrix with 2 columns. The first with pressure head values in [cm] and the second with hydraulic conductivity values log10[cm d-1].
- **parL** Defaults to NA has to be provided if weightmethod == "est1". See Details of (shypEstFun for explanation of parl)

**Details**

Character specifying weightmethod

- **user** user defined weights
- **none** no weights are considered, i.e. no measurement error assumed
- **range** rescaling (normalization of observations to the intervall [0,1]
- **fix1** fixed scalar weight for THETA is 0.05^2 and weight for log10K is 1
- **fix2** vector with the length of number of observations as given in retdata and condata are given, fixed to weight for THETA and condata
- **est1** Two scalar model weights (sigma^2-2) are treated as free parameters to be estimated by inversion, one for THETA and one for log10K

**Value**

The function returns a list of weights as specified through weightmethod

If weightmethod is set to est1 and parL is given as an extra argument, the function returns a list which is concatenated to the parL used in shypEstFun providing extra information on the nuisance parameters. Alternatively, parL can be passed as an argument to shypEstFun directly, accounting for the two additional nuisance parameters at the end of the respective vectors.

**Author(s)**

Tobias KD Weber
Examples

# Example 1 | fixed weights
weight.fix.L <- weightFun("fix1")

## Example 2 | range of measure data
data(shpdata1)

wrc <- shpdata1$TS1$wrc
hcc <- shpdata1$TS1$hcc
# Remove NAs
hcc <- shpdata1$TS1$hcc[!is.na(shpdata1$TS1$hcc[,1]) ,]
weight.fix.L <- weightFun("range", wrc, hcc)
Index

*Topic `Fredlund`
  shypFun, 37
*Topic `Kosugi`
  shypFun, 35
*Topic `Kvap`
  KvapFun, 8
  numMualem, 12
*Topic `PTF`
  logLikFun.norm, 10
  ptf.cW, 13
  ptf.vG2FM, 14
*Topic `VGM`
  shypFun, 27
  shypFun.01110, 31
  shypFun.01210, 32
  shypFun.01310, 34
  sncFun, 38
*Topic `fitting`
  gofFun, 5
  shypEstFun, 20
*Topic `initial pop`
  inipopFun, 7
*Topic `Kvap`
  numMualem, 12
*Topic `parameter estimation`
  spsh-package, 3
*Topic `resFun`
  resFun, 16
*Topic `soil hydrology`
  spsh-package, 3
*Topic `soil physics`
  spsh-package, 3
*Topic `transform`
  transBoundFun, 39
  transFun, 40
*Topic `weights`
  weightFun, 42
*Topic `datasets`
  shpdataQ, 18
  DEoptim.control, 22
gofFun, 5
inipopFun, 7
KvapFun, 8, 17, 22
lhs, 7
logLikFun.norm, 10
modMCMC, 22
numMualem, 12
optimumLHS, 7
ptf.cW, 13
ptf.vG2FM, 14
resFun, 16
shpdataQ, 18
shypEstFun, 4, 17, 20, 42
shypFun, 3, 4, 17, 20, 23, 27
shypFun.01110, 31
shypFun.01210, 32
shypFun.01310, 34
shypFun.02110, 35
shypFun.03110, 37
sncFun, 38
spsh (spsh-package), 3
spsh-package, 3
transBoundFun, 39
transFun, 40
weightFun, 42