**Package ‘BTYDplus’**

October 12, 2022

<table>
<thead>
<tr>
<th>Type</th>
<th>Package</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title</td>
<td>Probabilistic Models for Assessing and Predicting your Customer Base</td>
</tr>
<tr>
<td>Version</td>
<td>1.2.0</td>
</tr>
</tbody>
</table>

URL https://github.com/mplatzer/BTYDplus#readme

BugReports https://github.com/mplatzer/BTYDplus/issues

License GPL-3

LinkingTo Rcpp

Depends R (>= 3.2.0)

Imports Rcpp, BTYD (>= 2.3), coda, data.table, mvtnorm, bayesm, stats, graphics

Suggests testthat, covr, knitr, rmarkdown, gsl, lintr (>= 1.0.0)

RoxygenNote 7.1.1

LazyData true

VignetteBuilder knitr

NeedsCompilation yes

Author Michael Platzer [aut, cre]
## Maintainer
Michael Platzer <michael.platzer@gmail.com>

## Repository
CRAN

## Date/Publication
2021-01-21 11:00:16 UTC

### R topics documented:

<table>
<thead>
<tr>
<th>R Package</th>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>abe</td>
<td>GenerateData</td>
<td>3</td>
</tr>
<tr>
<td>abe.mcmc</td>
<td>DrawParameters</td>
<td>4</td>
</tr>
<tr>
<td>elog2cbs</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>elog2cum</td>
<td></td>
<td>7</td>
</tr>
<tr>
<td>estimateRegularity</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>groceryElog</td>
<td></td>
<td>9</td>
</tr>
<tr>
<td>mbgcnbd.cbs.LL</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>mbgcnbd.ConditionalExpectedTransactions</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>mbgcnbd.EstimateParameters</td>
<td></td>
<td>12</td>
</tr>
<tr>
<td>mbgcnbd.Expectation</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>mbgcnbd.ExpectedCumulativeTransactions</td>
<td></td>
<td>14</td>
</tr>
<tr>
<td>mbgcnbd.GenerateData</td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>mbgcnbd.PAlive</td>
<td></td>
<td>16</td>
</tr>
<tr>
<td>mbgcnbd.PlotFrequencyInCalibration</td>
<td></td>
<td>17</td>
</tr>
<tr>
<td>mbgcnbd.PlotFreqVsConditionalExpectedFrequency</td>
<td></td>
<td>19</td>
</tr>
<tr>
<td>mbgcnbd.PlotRecVsConditionalExpectedFrequency</td>
<td></td>
<td>20</td>
</tr>
<tr>
<td>mbgcnbd.PlotTrackingCum</td>
<td></td>
<td>21</td>
</tr>
<tr>
<td>mbgcnbd.PlotTrackingInc</td>
<td></td>
<td>23</td>
</tr>
<tr>
<td>mbgcnbd.pmf</td>
<td></td>
<td>25</td>
</tr>
<tr>
<td>mcmc</td>
<td>DrawFutureTransactions</td>
<td>26</td>
</tr>
<tr>
<td>mcmc</td>
<td>Expectation</td>
<td>27</td>
</tr>
<tr>
<td>mcmc</td>
<td>ExpectedCumulativeTransactions</td>
<td>28</td>
</tr>
<tr>
<td>mcmc</td>
<td>PActive</td>
<td>29</td>
</tr>
<tr>
<td>mcmc</td>
<td>PAlive</td>
<td>30</td>
</tr>
<tr>
<td>mcmc</td>
<td>PlotFrequencyInCalibration</td>
<td>30</td>
</tr>
<tr>
<td>mcmc</td>
<td>plotPActiveDiagnostic</td>
<td>32</td>
</tr>
<tr>
<td>mcmc</td>
<td>PlotTrackingCum</td>
<td>32</td>
</tr>
<tr>
<td>mcmc</td>
<td>PlotTrackingInc</td>
<td>34</td>
</tr>
<tr>
<td>mcmc</td>
<td>pmf</td>
<td>36</td>
</tr>
<tr>
<td>mcmc</td>
<td>setBurnin</td>
<td>37</td>
</tr>
<tr>
<td>nbd.cbs.LL</td>
<td></td>
<td>37</td>
</tr>
<tr>
<td>nbd</td>
<td>ConditionalExpectedTransactions</td>
<td>38</td>
</tr>
<tr>
<td>nbd</td>
<td>EstimateParameters</td>
<td>39</td>
</tr>
<tr>
<td>nbd</td>
<td>GenerateData</td>
<td>39</td>
</tr>
<tr>
<td>nbd</td>
<td>LL</td>
<td>40</td>
</tr>
<tr>
<td>pggg</td>
<td>GenerateData</td>
<td>41</td>
</tr>
<tr>
<td>pggg.mcmc</td>
<td>DrawParameters</td>
<td>42</td>
</tr>
<tr>
<td>pggg</td>
<td>plotRegularityRateHeterogeneity</td>
<td>43</td>
</tr>
<tr>
<td>plotTimingPatterns</td>
<td></td>
<td>44</td>
</tr>
<tr>
<td>pnbd</td>
<td>GenerateData</td>
<td>45</td>
</tr>
<tr>
<td>pnbd.mcmc</td>
<td>DrawParameters</td>
<td>46</td>
</tr>
</tbody>
</table>
abe.GenerateData  

Simulate data according to Pareto/NBD (Abe) model assumptions

Description
Simulate data according to Pareto/NBD (Abe) model assumptions

Usage
abe.GenerateData(
  n,  
  T.cal,  
  T.star,  
  params,  
  date.zero = "2000-01-01",  
  covariates = NULL  
)

Arguments
  n  Number of customers.
  T.cal  Length of calibration period. If a vector is provided, then it is assumed that 
         customers have different 'birth' dates, i.e. \( \max(T.cal) - T.cal \).
  T.star  Length of holdout period. This may be a vector.
  params  A list of model parameters: \( \beta \) and \( \gamma \).
  date.zero  Initial date for cohort start. Can be of class character, Date or POSIXt.
  covariates  Provide matrix of customer covariates. If NULL then random covariate values 
              between \([-1,1]\) are drawn.

Value
List of length 2:
  cbs  A data.frame with a row for each customer and the summary statistic as columns.
  elog  A data.frame with a row for each transaction, and columns cust, date and t.

Examples
# generate artificial Pareto/NBD (Abe) with 2 covariates
params <- list()
params$beta <- matrix(c(0.18, -2.5, 0.5, -0.3, -0.2, 0.8), byrow = TRUE, ncol = 2)
params$gamma <- matrix(c(0.05, 0.1, 0.1, 0.2), ncol = 2)
data <- abe.GenerateData(n = 200, T.cal = 32, T.star = 32, params)
cbs <- data$cbs  # customer by sufficient summary statistic - one row per customer
elog <- data$elog  # Event log - one row per event/purchase
abe.mcmc.DrawParameters

Pareto/NBD (Abe) Parameter Draws

Description

Returns draws from the posterior distributions of the Pareto/NBD (Abe) parameters, on cohort as well as on customer level.

Usage

abe.mcmc.DrawParameters(
  cal.cbs,
  covariates = c(),
  mcmc = 2500,
  burnin = 500,
  thin = 50,
  chains = 2,
  mc.cores = NULL,
  trace = 100
)

Arguments

cal.cbs Caliberation period customer-by-sufficient-statistic (CBS) data.frame. It must contain a row for each customer, and columns x for frequency, t.x for recency and T.cal for the total time observed. A correct format can be easily generated based on the complete event log of a customer cohort with elog2cbs.

covariates A vector of columns of cal.cbs which contain customer-level covariates.
mcmc Number of MCMC steps.
burnin Number of initial MCMC steps which are discarded.
thin Only every thin-th MCMC step will be returned.
chains Number of MCMC chains to be run.
mc.cores Number of cores to use in parallel (Unix only). Defaults to min(chains, detectCores()).
trace Print logging statement every trace-th iteration. Not available for mc.cores > 1.

Details

See demo(‘pareto-abe’) for how to apply this model.
elog2cbs

Value

List of length 2:

level_1 list of mcmc.lists, one for each customer, with draws for customer-level parameters $k, \lambda, \tau, z, \mu$

level_2 mcmc.list, with draws for cohort-level parameters

References


See Also

abe.GenerateData mcmc.PAlive mcmc.DrawFutureTransactions

Examples

data("groceryElog")
cbs <- elog2cbs(groceryElog, T.cal = "2006-12-31")
cbs$cov1 <- as.integer(cbs$cust) %% 2 # create dummy covariate
param.draws <- abe.mcmc.DrawParameters(cbs, c("cov1"),
    mcmc = 100, burnin = 50, thin = 10, chains = 1) # short MCMC to run demo fast

# cohort-level parameter draws
as.matrix(param.draws$level_2)
# customer-level parameter draws for customer with ID '4'
as.matrix(param.draws$level_1["4"])

# estimate future transactions
xstar.draws <- mcmc.DrawFutureTransactions(cbs, param.draws, cbs$T.star)
xstar.est <- apply(xstar.draws, 2, mean)
head(xstar.est)

elog2cbs

Convert Event Log to customer-level summary statistic

desc

Efficient implementation for the conversion of an event log into a customer-by-sufficient-statistic (CBS) data.frame, with a row for each customer, which is the required data format for estimating model parameters.

Usage
elog2cbs(elog, units = "week", T.cal = NULL, T.tot = NULL)
Arguments

elog  Event log, a data.frame with field cust for the customer ID and field date for the date/time of the event, which should be of type Date or POSIXt. If a field sales is present, it will be aggregated as well.

units  Time unit, either week, day, hour, min or sec. See difftime.

T.cal  End date of calibration period. Defaults to max(elog$date).

T.tot  End date of the observation period. Defaults to max(elog$date).

Details

The time unit for expressing t.x, T.cal and litt are determined via the argument units, which is passed forward to method difftime, and defaults to weeks.

Argument T.tot allows one to specify the end of the observation period, i.e. the last possible date of an event to still be included in the event log. If T.tot is not provided, then the date of the last recorded event will be assumed to coincide with the end of the observation period. If T.tot is provided, then any event that occurs after that date is discarded.

Argument T.cal allows one to split the summary statistics into a calibration and a holdout period. This can be useful for evaluating forecasting accuracy for a given dataset. If T.cal is not provided, then the whole observation period is considered, and is then subsequently used for for estimating model parameters. If it is provided, then the returned data.frame contains two additional fields, with x.star representing the number of repeat transactions during the holdout period of length T.star. And only those customers are contained, who have had at least one event during the calibration period.

Transactions with identical cust and date field are treated as a single transaction, with sales being summed up.

Value

data.frame with fields:

cust  Customer id (unique key).

x  Number of recurring events in calibration period.

t.x  Time between first and last event in calibration period.

litt  Sum of logarithmic intertransaction timings during calibration period.

sales  Sum of sales in calibration period, incl. initial transaction. Only if elog$sales is provided.

sales.x  Sum of sales in calibration period, excl. initial transaction. Only if elog$sales is provided.

first  Date of first transaction in calibration period.

T.cal  Time between first event and end of calibration period.

T.star  Length of holdout period. Only if T.cal is provided.

x.star  Number of events within holdout period. Only if T.cal is provided.

sales.star  Sum of sales within holdout period. Only if T.cal and elog$sales are provided.
**elog2cum**

**Convert Event Log to Transaction Counts**

**Description**
Aggregates an event log to either incremental or cumulative number of transactions. If first=TRUE then the initial transactions of each customer are included in the count as well.

**Usage**

```r
elog2cum(elog, by = 7, first = FALSE)
```

```r
elog2inc(elog, by = 7, first = FALSE)
```

**Arguments**

- **elog**  
  Event log, a data.frame with columns cust and transaction time t or date.

- **by**  
  Only return every by-th count Defaults to 7, and thus returns weekly numbers.

- **first**  
  If TRUE, then the first transaction for each customer is being counted as well.

**Details**
Duplicate transactions with identical cust and date (or t) field are counted only once.

**Value**
Numeric vector of transaction counts.

**Examples**

```r
data("groceryElog")
cum <- elog2cum(groceryElog)
plot(cum, typ="1", frame = FALSE)
inc <- elog2inc(groceryElog)
plot(inc, typ="1", frame = FALSE)
```
estimateRegularity  Estimate Regularity in Intertransaction Timings

Description

The models (M)BG/CNBD-k and Pareto/GGG are capable of leveraging regularity within transaction timings for improving forecast accuracy. This method provides a quick check for the degree of regularity in the event timings. A return value of close to 1 supports the assumption of exponentially distributed intertransaction times, whereas values significantly larger than 1 reveal the presence of regularity.

Usage

```
estimateRegularity(
  elog,
  method = "wheat",
  plot = FALSE,
  title = "",
  min = NULL
)
```

Arguments

- `elog`: Event log, a `data.frame` with columns `cust` and transaction time `t` or `date`.
- `method`: Either `wheat`, `mle`, `mle-minka`, `mle-thom` or `cv`.
- `plot`: If `TRUE` then an additional diagnostic plot is provided.
- `title`: Plot title.
- `min`: Minimum number of intertransaction times per customer. Customers with less than `min` intertransactions are not considered. Defaults to 2 for method ‘wheat’, and to 10 otherwise.

Details

Estimation is either done by 1) assuming the same degree of regularity across all customers (Wheat & Morrison (1990) via method = "wheat"), or 2) by estimating regularity for each customer separately, as the shape parameter of a fitted gamma distribution, and then return the median across estimates. The latter methods, though, require sufficient (\(\geq\) min) transactions per customer.

Wheat & Morrison (1990)’s method calculates for each customer a statistic \(M\) based on her last two number of intertransaction times as \(\frac{ITT_1}{(ITT_1 + ITT_2)}\). That measure is known to follow a Beta(\(k\), \(k\)) distribution, and \(k\) can be estimated as \(\frac{(1-4\text{Var}(M))}{8\text{Var}(M)}\). The corresponding diagnostic plot (plot = TRUE) shows the actual distribution of \(M\) vs. the theoretical distribution for \(k = 1\) and \(k = 2\).

Value

Estimated real-valued regularity parameter.
References


Examples

data("groceryElog")
estimateRegularity(groceryElog, plot = TRUE, method = 'wheat')
estimateRegularity(groceryElog, plot = TRUE, method = 'mle-minka')
estimateRegularity(groceryElog, plot = TRUE, method = 'mle-thom')
estimateRegularity(groceryElog, plot = TRUE, method = 'cv')

---

groceryElog

Event log for customers of an online grocery store.

Description

These data came from an online retailer offering a broad range of grocery categories. The original data set spans four years, but lacked the customers’ acquisition date. Therefore, we constructed a quasi cohort by limiting the provided data analysis to those customers who haven’t purchased at all in the first two years, and had their first purchase in the first quarter of 2006. This resulted in 10483 transactions being recorded for 1525 customers during a period of two years (2006-2007).

Usage

groceryElog

Format

A data frame with 10483 rows and 2 variables:

cust customer ID, factor vector
date transaction date, Date vector

Source

Thomas Reutterer <thomas.reutterer@wu.ac.at>

References

**mbgcnbd.cbs.LL**

(M)BG/CNBD-k Log-Likelihood

**Description**

Calculates the log-likelihood of the (M)BG/CNBD-k model.

**Usage**

```r
mbgcnbd.cbs.LL(params, cal.cbs)
mbgcnbd.LL(params, x, t.x, T.cal, litt)
bgcnbd.cbs.LL(params, cal.cbs)
bgcnbd.LL(params, x, t.x, T.cal, litt)
```

**Arguments**

- `params`: A vector with model parameters `k`, `r`, `alpha`, `a`, and `b`, in that order.
- `cal.cbs`: Calibration period customer-by-sufficient-statistic (CBS) data.frame. It must contain a row for each customer, and columns `x` for frequency, `t.x` for recency, `T.cal` for the total time observed, as well as the sum over logarithmic inter-transaction times `litt`. A correct format can be easily generated based on the complete event log of a customer cohort with `elog2cbs`.
- `x`: frequency, i.e. number of re-purchases
- `t.x`: recency, i.e. time elapsed from first purchase to last purchase
- `T.cal`: total time of observation period
- `litt`: sum of logarithmic interpurchase times

**Value**

For `bgcnbd.cbs.LL`, the total log-likelihood of the provided data. For `bgcnbd.LL`, a vector of log-likelihoods as long as the longest input vector (`x`, `t.x`, or `T.cal`).

**References**

Description

Uses (M)BG/CNBD-k model parameters and a customer’s past transaction behavior to return the number of transactions they are expected to make in a given time period.

Usage

\[
\text{mbgcnbd.\text{ConditionalExpectedTransactions}}(\text{params, T.star, x, t.x, T.cal})
\]

\[
\text{bgcnbd.\text{ConditionalExpectedTransactions}}(\text{params, T.star, x, t.x, T.cal})
\]

Arguments

- **params**: A vector with model parameters \(k, r, \alpha, a\) and \(b\), in that order.
- **T.star**: Length of time for which we are calculating the expected number of transactions.
- **x**: Number of repeat transactions in the calibration period \(T.cal\), or a vector of calibration period frequencies.
- **t.x**: Recency, i.e., length between first and last transaction during calibration period.
- **T.cal**: Length of calibration period, or a vector of calibration period lengths.

Value

Number of transactions a customer is expected to make in a time period of length \(t\), conditional on their past behavior. If any of the input parameters has a length greater than 1, this will be a vector of expected number of transactions.

References


Examples

```r
## Not run:
data("groceryElog")
cbs <- elog2cbs(groceryElog)
params <- mbgcnbd.EstimateParameters(cbs, k = 2)
# estimate transactions for next 12 weeks
xstar.est <- mbgcnbd.\text{ConditionalExpectedTransactions}(\text{params, T.star = 12, cbs}\$x, cbs}\$t.x, cbs}\$T.cal)
head(xstar.est) # expected number of transactions for first 6 customers
sum(xstar.est) # expected total number of transactions during holdout
```
mbgcnbd.EstimateParameters

(M)BG/CNBD-k Parameter Estimation

Description

Estimates parameters for the (M)BG/CNBD-k model via Maximum Likelihood Estimation.

Usage

mbgcnbd.EstimateParameters(
  cal.cbs,
  k = NULL,
  par.start = c(1, 3, 1, 3),
  max.param.value = 10000,
  trace = 0
)

bgcnbd.EstimateParameters(
  cal.cbs,
  k = NULL,
  par.start = c(1, 3, 1, 3),
  max.param.value = 10000,
  trace = 0
)

mbgnbd.EstimateParameters(
  cal.cbs,
  par.start = c(1, 3, 1, 3),
  max.param.value = 10000,
  trace = 0
)

Arguments

cal.cbs Calibration period customer-by-sufficient-statistic (CBS) data.frame. It must contain a row for each customer, and columns x for frequency, t.x for recency, T.cal for the total time observed, as well as the sum over logarithmic intertransaction times litt, in case that k is not provided. A correct format can be easily generated based on the complete event log of a customer cohort with elog2cbs.

k Integer-valued degree of regularity for Erlang-k distributed interpurchase times. By default this k is not provided, and a grid search from 1 to 12 is performed in order to determine the best-fitting k. The grid search is stopped early, if the log-likelihood does not increase anymore when increasing k beyond 4.
par.start  
Initial (M)BG/CNBD-k parameters. A vector with \( r, \alpha, a \) and \( b \) in that order.

max.param.value  
Upper bound on parameters.

trace  
If larger than 0, then the parameter values are printed every \( \text{trace} \)-step of the maximum likelihood estimation search.

Value  
A vector of estimated parameters.

References  


See Also  
\texttt{bgnbd.EstimateParameters}

Examples  
```r
## Not run:
data("groceryElog")
cbs <- elog2cbs(groceryElog)
$params <- mbgcnbd.EstimateParameters(cbs)

## End(Not run)
```

Description  
Returns the number of repeat transactions that a randomly chosen customer (for whom we have no prior information) is expected to make in a given time period, i.e. \( E(X(t)|k, r, \alpha, a, b) \).

Usage  
\texttt{mbgcnbd.Expectation(params, t)}

\texttt{bgnbd.Expectation(params, t)}
mbgcncbd.ExpectedCumulativeTransactions

**Arguments**

- **params**: A vector with model parameters $k$, $r$, $\alpha$, $a$ and $b$, in that order.
- **t**: Length of time for which we are calculating the expected number of repeat transactions.

**Details**

Note: Computational time increases with the number of unique values of $t$.

**Value**

Number of repeat transactions a customer is expected to make in a time period of length $t$.

**References**


**Examples**

```r
## Not run:
data("groceryElog")
cbs <- elog2cbs(groceryElog)
params <- mbgcncbd.EstimateParameters(cbs)
mbgcncbd.Expectation(params, t = c(26, 52))

## End(Not run)
```

---

mbgcncbd.ExpectedCumulativeTransactions

*(M)BG/CNBD-k Expected Cumulative Transactions*

**Description**

Calculates the expected cumulative total repeat transactions by all customers for the calibration and holdout periods.

**Usage**

mbgcncbd.ExpectedCumulativeTransactions(params, T.cal, T.tot, n.periods.final)

bgcncbd.ExpectedCumulativeTransactions(params, T.cal, T.tot, n.periods.final)
Arguments

- params: A vector with model parameters $k$, $r$, $\alpha$, $a$ and $b$, in that order.
- T.cal: A vector to represent customers’ calibration period lengths.
- T.tot: End of holdout period. Must be a single value, not a vector.
- n.periods.final: Number of time periods in the calibration and holdout periods.

Details

Note: Computational time increases with the number of unique values of `T.cal`.

Value

Vector of length `n.periods.final` with expected cumulative total repeat transactions by all customers.

Examples

```r
## Not run:
data("groceryElog")
cbs <- elog2cbs(groceryElog)
params <- mbgcnbd.EstimateParameters(cbs, k = 2)
# Returns a vector containing expected cumulative repeat transactions for 104
# weeks, with every eighth week being reported.
mbgcnbd.ExpectedCumulativeTransactions(params,
  T.cal = cbs$T.cal,
  T.tot = 104,
  n.periods.final = 104 / 8)
## End(Not run)
```
Arguments

n  Number of customers.
T.cal  Length of calibration period. If a vector is provided, then it is assumed that customers have different ‘birth’ dates, i.e. \( \max(T.\text{cal}) - T.\text{cal} \).
T.star  Length of holdout period. This may be a vector.
params  A vector with model parameters k, r, alpha, a and b, in that order.
date.zero  Initial date for cohort start. Can be of class character, Date or POSIXt.

Value

List of length 2:
cbs  A data.frame with a row for each customer and the summary statistic as columns.
elog  A data.frame with a row for each transaction, and columns cust, date and t.

References


Examples

params <- c(k = 3, r = 0.85, alpha = 1.45, a = 0.79, b = 2.42)
data <- mbgcnbd.GenerateData(n = 200, T.cal = 24, T.star = 32, params)

# customer by sufficient summary statistic - one row per customer
head(data$cbs)

# event log - one row per event/transaction
head(data$elog)

mbgcnbd.PAlive  \((M)BG/CNBD-k P(\text{alive})\)

Description

Uses (M)BG/CNBD-k model parameters and a customer’s past transaction behavior to return the probability that they are still alive at the end of the calibration period.

Usage

mbgcnbd.PAlive(params, x, t.x, T.cal)
bgcnbd.PAlive(params, x, t.x, T.cal)
Arguments

params  A vector with model parameters k, r, alpha, a and b, in that order.
x      Number of repeat transactions in the calibration period T.cal, or a vector of calibration period frequencies.
t.x  Recency, i.e. length between first and last transaction during calibration period.
T.cal Length of calibration period, or a vector of calibration period lengths.

Value

Probability that the customer is still alive at the end of the calibration period.

References


Examples

```r
## Not run:
data("groceryElog")
cbs <- elog2cbs(groceryElog)
params <- mbgcnbd.EstimateParameters(cbs)
palive <- mbgcnbd.PAlive(params, cbs$x, cbs$t.x, cbs$T.cal)
head(palive) # Probability of being alive for first 6 customers
mean(palive) # Estimated share of customers to be still alive
## End(Not run)
```

mbgcnbd.PlotFrequencyInCalibration

(M)BG/CNBD-k Plot Frequency in Calibration Period

Description

Plots a histogram and returns a matrix comparing the actual and expected number of customers who made a certain number of repeat transactions in the calibration period, binned according to calibration period frequencies.

Usage

```r
mbgcnbd.PlotFrequencyInCalibration(
  params,
  cal.cbs,
  censor = 7,
  xlab = "Calibration period transactions",
  ylab = "Customers",
)```
title = "Frequency of Repeat Transactions"
)

bgcnbd.PlotFrequencyInCalibration(
  params,
  cal.cbs,
  censor = 7,
  xlab = "Calibration period transactions",
  ylab = "Customers",
  title = "Frequency of Repeat Transactions"
)

Arguments

params A vector with model parameters k, r, alpha, a and b, in that order.

cal.cbs Calibration period CBS (customer by sufficient statistic). It must contain columns
  for frequency ('x') and total time observed ('T.cal').

censor Cutoff point for number of transactions in plot.

xlab Descriptive label for the x axis.

ylab Descriptive label for the y axis.

title Title placed on the top-center of the plot.

Value

Calibration period repeat transaction frequency comparison matrix (actual vs. expected).

References

(M)BG/CNBD-k: Reutterer, T., Platzer, M., & Schroeder, N. (2020). Leveraging purchase regular-
ity for predicting customer behavior the easy way. International Journal of Research in Marketing.

Examples

## Not run:
data("groceryElog")
cbs <- elog2cbs(groceryElog)
params <- mbgcnbd.EstimateParameters(cbs)
bgcnbd.PlotFrequencyInCalibration(params, cbs)
## End(Not run)
mbgcnbd.PlotFreqVsConditionalExpectedFrequency

(M)BG/CNBD-k Plot Frequency vs. Conditional Expected Frequency

Description

Plots the actual and conditional expected number transactions made by customers in the holdout period, binned according to calibration period frequencies, and returns this comparison in a matrix.

Usage

mbgcnbd.PlotFreqVsConditionalExpectedFrequency(
  params,
  T.star,
  cal.cbs,
  x.star,
  censor,
  xlab = "Calibration period transactions",
  ylab = "Holdout period transactions",
  xticklab = NULL,
  title = "Conditional Expectation"
)

Arguments

- **params**: A vector with model parameters \( k, r, \alpha, a \) and \( b \), in that order.
- **T.star**: Length of the holdout period.
- **cal.cbs**: Calibration period CBS (customer by sufficient statistic). It must contain columns for frequency (\( x \)), recency (\( t.x \)) and total time observed (\( T.cal \)).
- **x.star**: Vector of transactions made by each customer in the holdout period.
- **censor**: Cutoff point for number of transactions in plot.
- **xlab**: Descriptive label for the x axis.
- **ylab**: Descriptive label for the x axis.
mbgcndb.PlotRecVsConditionalExpectedFrequency

xticklab A vector containing a label for each tick mark on the x axis.
title Title placed on the top-center of the plot.

Value

Holdout period transaction frequency comparison matrix (actual vs. expected).

See Also

mbgcndb.PlotFreqVsConditionalExpectedFrequency

Examples

## Not run:
data("groceryElog")
cbs <- elog2cbs(groceryElog, T.cal = "2006-09-30")
params <- mbgcndb.EstimateParameters(cbs, k=2)
mbgcndb.PlotFreqVsConditionalExpectedFrequency(params, T.star=52, cbs, cbs$x.star, censor=7)
## End(Not run)

mbgcndb.PlotRecVsConditionalExpectedFrequency

(M)BG/CNBD-k Plot Actual vs. Conditional Expected Frequency by Recency

Description

Plots the actual and conditional expected number transactions made by customers in the holdout period, binned according to calibration period recencies, and returns this comparison in a matrix.

Usage

mbgcndb.PlotRecVsConditionalExpectedFrequency(
  params,
  cal.cbs,
  T.star,
  x.star,
  xlab = "Calibration period recency",
  ylab = "Holdout period transactions",
  xticklab = NULL,
  title = "Actual vs. Conditional Expected Transactions by Recency"
)

bgcndb.PlotRecVsConditionalExpectedFrequency(
  params,
  cal.cbs,
  T.star,
Arguments

- **params**: A vector with model parameters $k$, $r$, $\alpha$, $a$, and $b$, in that order.
- **cal.cbs**: Calibration period CBS (customer by sufficient statistic). It must contain columns for frequency ('x'), recency ('t.x') and total time observed ('T.cal').
- **T.star**: Length of the holdout period.
- **x.star**: Vector of transactions made by each customer in the holdout period.
- **xlab**: Descriptive label for the x axis.
- **ylab**: Descriptive label for the x axis.
- **xticklab**: A vector containing a label for each tick mark on the x axis.
- **title**: Title placed on the top-center of the plot.

Value

Matrix comparing actual and conditional expected transactions in the holdout period.

See Also

- `bgcnbd.PlotFreqVsConditionalExpectedFrequency`

Examples

```r
## Not run:
data("groceryElog")
cbs <- elog2cbs(groceryElog, T.cal = "2006-09-30")
params <- mbgcnbd.EstimateParameters(cbs, k=2)
mbgcnbd.PlotRecVsConditionalExpectedFrequency(params, cbs, T.star=52, cbs$x.star)
## End(Not run)
```
Usage

mbgcnbd.PlotTrackingCum(
  params,
  T.cal,
  T.tot,
  actual.cu.tracking.data,
  xlab = "Week",
  ylab = "Cumulative Transactions",
  xticklab = NULL,
  title = "Tracking Cumulative Transactions",
  ymax = NULL,
  legend = c("Actual", "Model")
)

Arguments

params A vector with model parameters k, r, alpha, a and b, in that order.
T.cal A vector to represent customers’ calibration period lengths.
T.tot End of holdout period. Must be a single value, not a vector.
actual.cu.tracking.data A vector containing the cumulative number of repeat transactions made by customers for each period in the total time period (both calibration and holdout periods).
xlab Descriptive label for the x axis.
ylab Descriptive label for the y axis.
xticklab A vector containing a label for each tick mark on the x axis.
title Title placed on the top-center of the plot.
ymax Upper boundary for y axis.
legend plot legend, defaults to ‘Actual’ and ‘Model’.

Details

Note: Computational time increases with the number of unique values of T.cal.
### mbgcnbd.PlotTrackingInc

**Value**

Matrix containing actual and expected cumulative repeat transactions.

**See Also**

mbgcnbd.ExpectedCumulativeTransactions

**Examples**

```r
## Not run:
data("groceryElog")
groceryElog <- groceryElog[groceryElog$date < "2006-06-30", ]
cbs <- elog2cbs(groceryElog, T.cal = "2006-04-30")
cum <- elog2cum(groceryElog)
params <- mbgcnbd.EstimateParameters(cbs, k = 2)
mbgcnbd.PlotTrackingCum(params, cbs$T.cal,
    T.tot = max(cbs$T.cal + cbs$T.star), cum)
## End(Not run)
```

### mbgcnbd.PlotTrackingInc

*(M)BG/CNBD-k Tracking Incremental Transactions Comparison*

**Description**

Plots the actual and expected incremental total repeat transactions by all customers for the calibration and holdout periods, and returns this comparison in a matrix.

**Usage**

```r
mbgcnbd.PlotTrackingInc(
    params,
    T.cal,
    T.tot,
    actual.inc.tracking.data,
    xlab = "Week",
    ylab = "Transactions",
    xticklab = NULL,
    title = "Tracking Weekly Transactions",
    ymax = NULL,
    legend = c("Actual", "Model")
)
```

```r
bgcnbd.PlotTrackingInc(
    params,
    T.cal,
    T.tot,
```
actual.inc.tracking.data,
xlab = "Week",
ylab = "Transactions",
xticklab = NULL,
title = "Tracking Weekly Transactions",
ymax = NULL,
legend = c("Actual", "Model")
)

Arguments

params A vector with model parameters k, r, alpha, a and b, in that order.
T.cal A vector to represent customers' calibration period lengths.
T.tot End of holdout period. Must be a single value, not a vector.
actual.inc.tracking.data A vector containing the incremental number of repeat transactions made by customers for each period in the total time period (both calibration and holdout periods).
xlab Descriptive label for the x axis.
ylab Descriptive label for the y axis.
xticklab A vector containing a label for each tick mark on the x axis.
title Title placed on the top-center of the plot.
ymax Upper boundary for y axis.
legend plot legend, defaults to 'Actual' and 'Model'.

Details

Note: Computational time increases with the number of unique values of T.cal.

Value

Matrix containing actual and expected incremental repeat transactions.

See Also

mbgcnbd.expectedCumulativeTransactions

Examples

## Not run:
data("groceryElog")
groceryElog <- groceryElog[groceryElog$date < "2006-06-30", ]
cbs <- elog2cbs(groceryElog, T.cal = "2006-04-30")
inc <- elog2inc(groceryElog)
params <- mbgcnbd.EstimateParameters(cbs, k = 2)
mbgcnbd.PlotTrackingInc(params, cbs$T.cal, T.tot = max(cbs$T.cal + cbs$T.star), inc)
## End(Not run)
Description

Uses (M)BG/CNBD-k model parameters to return the probability distribution of purchase frequencies for a random customer in a given time period, i.e. \( P(X(t) = x | r, \alpha, a, b) \).

Usage

\[
\text{mbgcnbd.pmf}(\text{params, t, x}) \\
\text{bgcnbd.pmf}(\text{params, t, x})
\]

Arguments

- **params**: A vector with model parameters \( k, r, \alpha, a \) and \( b \), in that order.
- **t**: Length end of time period for which probability is being computed. May also be a vector.
- **x**: Number of repeat transactions for which probability is calculated. May also be a vector.

Value

\( P(X(t) = x | r, \alpha, a, b) \). If either \( t \) or \( x \) is a vector, then the output will be a vector as well. If both are vectors, the output will be a matrix.

References


Examples

```r
## Not run:
data("groceryElog")
cbs <- elog2cbs(groceryElog)
params <- mbgcnbd.EstimateParameters(cbs)
mbgcnbd.pmf(params, t = 52, x = 0:6)
mbgcnbd.pmf(params, t = c(26, 52), x = 0:6)
## End(Not run)
```
mcmc.DrawFutureTransactions

*Draws number of future transactions based on MCMC parameter draws*

**Description**

For each customer and each provided MCMC parameter draw this method will sample the number of transactions during the holdout period $T_{\text{star}}$. If argument `size` is provided then it returns a flexible number of draws, whereas for each customer and each draw it will first make a draw from the parameter draws.

**Usage**

```r
mcmc.DrawFutureTransactions(
  cal.cbs,
  draws,
  $T_{\text{star}}$ = cal.cbs$T_{\text{star}},
  sample_size = NULL
)
```

**Arguments**

- `cal.cbs`: Calibration period customer-by-sufficient-statistic (CBS) data.frame.
- `draws`: MCMC draws as returned by `*.mcmc.DrawParameters`
- `T_{\text{star}}`: Length of period for which future transactions are counted.
- `sample_size`: Number of samples to draw. Defaults to the same number of parameter draws that are passed to `draws`.

**Value**

2-dim matrix [draw x customer] with sampled future transactions.

**Examples**

```r
data("groceryElog")
cbs <- elog2cbs(groceryElog, T.cal = "2006-12-31")
param.draws <- pnbd.mcmc.DrawParameters(cbs,
  mcmc = 100, burnin = 50, thin = 10, chains = 1) # short MCMC to run demo fast
xstar.draws <- mcmc.DrawFutureTransactions(cbs, param.draws)
cbs$xstar.est <- apply(xstar.draws, 2, mean)
cbs$pactive <- mcmc.PActive(xstar.draws)
head(cbs)
```
### mcmc.Expectation

*Unconditional Expectation for Pareto/GGG, Pareto/NBD (HB) and Pareto/NBD (Abe)*

#### Description

Uses model parameter draws to return the expected number of repeat transactions that a randomly chosen customer (for whom we have no prior information) is expected to make in a given time period.

\[ E(X(t)) \]

#### Usage

```r
mcmc.Expectation(draws, t, sample_size = 10000)
```

#### Arguments

- `draws`: MCMC draws as returned by `*.mcmc.DrawParameters`
- `t`: Length of time for which we are calculating the expected number of transactions. May also be a vector.
- `sample_size`: Sample size for estimating the probability distribution.

#### Details

The expected transactions need to be sampled. Due to this sampling, the return result varies from one call to another. Larger values of `sample_size` will generate more stable results.

#### Value

Number of repeat transactions a customer is expected to make in a time period of length `t`.

#### Examples

```r
data("groceryElog")
cbs <- elog2cbs(groceryElog)
param.draws <- pnbd.mcmc.DrawParameters(cbs,
    mcmc = 100, burnin = 50, thin = 10, chains = 1) # short MCMC to run demo fast
mcmc.Expectation(param.draws, t = c(26, 52))
```
mcmc.ExpectedCumulativeTransactions

Expected Cumulative Transactions for Pareto/GGG, Pareto/NBD (HB) and Pareto/NBD (Abe)

Description

Uses model parameter draws to return the expected number of repeat transactions that a randomly chosen customer (for whom we have no prior information) is expected to make in a given time period.

Usage

mcmc.ExpectedCumulativeTransactions(
  draws,
  T.cal,
  T.tot,
  n.periods.final,
  sample_size = 10000,
  covariates = NULL
)

Arguments

draws MCMC draws as returned by *mcmc.DrawParameters
T.cal A vector to represent customers’ calibration period lengths (in other words, the T.cal column from a customer-by-sufficient-statistic matrix). Considering rounding in order to speed up calculations.
T.tot End of holdout period. Must be a single value, not a vector.
n.periods.final Number of time periods in the calibration and holdout periods.
sample_size Sample size for estimating the probability distribution.
covariates (optional) Matrix of covariates, for Pareto/NBD (Abe) model, passed to abe.GenerateData for simulating data.

Details

The expected transactions need to be sampled. Due to this sampling, the return result varies from one call to another. Larger values of sample_size will generate more stable results.

Value

Numeric vector of expected cumulative total repeat transactions by all customers.
Examples

data("groceryElog")
cbs <- elog2cbs(groceryElog)
param.draws <- pnbd.mcmc.DrawParameters(cbs,
  mcmc = 100, burnin = 50, thin = 10, chains = 1)  # short MCMC to run demo fast
# Returns a vector containing expected cumulative repeat transactions for 104
# weeks, with every eighth week being reported.
mcmc EXPECTED CUMULATIVE TRANSACTIONS (PARAM.DRAWS,
  T.cal = cbs$T.cal, T.tot = 104, n.periods.final = 104/8, sample.size = 1000)

mcmc.PActive

Calculates P(active) based on drawn future transactions.

Description

Calculates P(active) based on drawn future transactions.

Usage

  mcmc.PActive(xstar)

Arguments

  xstar  Future transaction draws as returned by mcmc.DrawFutureTransactions.

Value

  numeric A vector with the customers’ probabilities of being active during the holdout period.

Examples

data("groceryElog")
cbs <- elog2cbs(groceryElog, T.cal = "2006-12-31")
param.draws <- pnbd.mcmc.DrawParameters(cbs,
  mcmc = 100, burnin = 50, thin = 10, chains = 1)  # short MCMC to run demo fast
xstar.draws <- mcmc.DrawFutureTransactions(cbs, param.draws)
cbs$pactive <- mcmc.PActive(xstar.draws)
head(cbs)
mcmc.PAlive

Calculates \( P(\text{alive}) \) based on MCMC parameter draws

Description

Calculates \( P(\text{alive}) \) based on MCMC parameter draws

Usage

mcmc.PAlive(draws)

Arguments

draws MCMC draws as returned by *.mcmc.DrawParameters

Value

Numeric vector with the customers' probabilities of being still alive at end of calibration period

Examples

data("groceryElog")
cbs <- elog2cbs(groceryElog)
param.draws <- pnbd.mcmc.DrawParameters(cbs,
    mcmc = 100, burnin = 50, thin = 10, chains = 1) # short MCMC to run demo fast
palive <- mcmc.PAlive(param.draws)
head(palive)
mean(palive)

mcmc.PlotFrequencyInCalibration

Frequency in Calibration Period for Pareto/GGG, Pareto/NBD (HB) and Pareto/NBD (Abe)

Description

Plots a histogram and returns a matrix comparing the actual and expected number of customers who made a certain number of repeat transactions in the calibration period, binned according to calibration period frequencies.
Usage

```r
mcmc.PlotFrequencyInCalibration(
  draws, 
  cal.cbs, 
  censor = 7, 
  xlab = "Calibration period transactions", 
  ylab = "Customers", 
  title = "Frequency of Repeat Transactions", 
  sample_size = 1000 
)
```

Arguments

- **draws**: MCMC draws as returned by `*.mcmc.DrawParameters`
- **cal.cbs**: Calibration period customer-by-sufficient-statistic (CBS) data.frame. It must contain columns for frequency (`'x'`) and total time observed (`'T.cal'`).
- **censor**: Cutoff point for number of transactions in plot.
- **xlab**: Descriptive label for the x axis.
- **ylab**: Descriptive label for the y axis.
- **title**: Title placed on the top-center of the plot.
- **sample_size**: Sample size for estimating the probability distribution. See `mcmc.pmf`.

Details

The method `mcmc.pmf` is called to calculate the expected numbers based on the corresponding model.

Value

Calibration period repeat transaction frequency comparison matrix (actual vs. expected).

See Also

`mcmc.pmf`

Examples

```r
## Not run:
data("groceryElog")
cbs <- elog2cbs(groceryElog, T.cal = "2006-12-31")
param.draws <- pnbd.mcmc.DrawParameters(cbs,
  mcmc = 100, burnin = 50, thin = 10, chains = 1) # short MCMC to run demo fast
mcmc.PlotFrequencyInCalibration(param.draws, cbs, sample_size = 1000)

## End(Not run)
```
mcmc.plotPActiveDiagnostic

Draw diagnostic plot to inspect error in \( P(\text{active}) \).

**Description**

Draw diagnostic plot to inspect error in \( P(\text{active}) \).

**Usage**

\[
mcmc.plotPActiveDiagnostic(cbs, xstar, title = "Diagnostic Plot for \( P(\text{active}) \))
\]

**Arguments**

- **cbs**: A data.frame with column \( x \) and \( x.\text{star} \).
- **xstar**: Future transaction draws as returned by `mcmc.DrawFutureTransactions`.
- **title**: Plot title.

**Examples**

```r
data("groceryElog")
cbs <- elog2cbs(groceryElog, T.cal = "2006-12-31")
param.draws <- pnbd.mcmc.DrawParameters(cbs,
    mcmc = 100, burnin = 50, thin = 10, chains = 1) # short MCMC to run demo fast
xstar.draws <- mcmc.DrawFutureTransactions(cbs, param.draws)
mcmc.plotPActiveDiagnostic(cbs, xstar.draws)
```

mcmc.PlotTrackingCum

*Tracking Cumulative Transactions Plot for Pareto/GGG, Pareto/NBD (HB) and Pareto/NBD (Abe)*

**Description**

Plots the actual and expected cumulative total repeat transactions by all customers for the calibration and holdout periods, and returns this comparison in a matrix.

**Usage**

\[
mcmc.PlotTrackingCum(
    draws,
    T.cal,
    T.tot,
    actual.cu.tracking.data,
    xlab = "Week",
    ylab = "Cumulative Transactions",
)
\]
mcmc.PlotTrackingCum

xticklab = NULL,
title = "Tracking Cumulative Transactions",
ymax = NULL,
sample_size = 10000,
covariates = NULL,
legend = c("Actual", "Model")
)

Arguments

draws MCMC draws as returned by *.mcmc.DrawParameters
T.cal A vector to represent customers’ calibration period lengths (in other words, the T.cal column from a customer-by-sufficient-statistic matrix). Considering rounding in order to speed up calculations.
T.tot End of holdout period. Must be a single value, not a vector.
actual.cu.tracking.data A vector containing the cumulative number of repeat transactions made by customers for each period in the total time period (both calibration and holdout periods).
xlab Descriptive label for the x axis.
ylab Descriptive label for the y axis.
xticklab A vector containing a label for each tick mark on the x axis.
title Title placed on the top-center of the plot.
ymax Upper boundary for y axis.
sample_size Sample size for estimating the probability distribution. See mcmc.ExpectedCumulativeTransactions.
covariates (optional) Matrix of covariates, for Pareto/NBD (Abe) model, passed to abe.GenerateData for simulating data.
legend plot legend, defaults to ‘Actual’ and ‘Model’.

Details

The expected transactions need to be sampled. Due to this sampling, the return result varies from one call to another. Larger values of sample_size will generate more stable results.

Value

Matrix containing actual and expected cumulative repeat transactions.

See Also

mcmc.PlotTrackingInc mcmc.ExpectedCumulativeTransactions elog2cum
Examples

```r
## Not run:
data("groceryElog")
cbs <- elog2cbs(groceryElog, T.cal = "2006-12-31")
cum <- elog2cum(groceryElog)
param.draws <- pnbd.mcmc.DrawParameters(cbs)
mat <- mcmc.PlotTrackingCum(param.draws,
  T.cal = cbs$T.cal,
  T.tot = max(cbs$T.cal + cbs$T.star),
  actual.cu.tracking.data = cum)
## End(Not run)
```

### mcmc.PlotTrackingInc

**Tracking Incremental Transactions Plot for Pareto/GGG, Pareto/NBD (HB) and Pareto/NBD (Abe)**

#### Description

Plots the actual and expected incremental total repeat transactions by all customers for the calibration and holdout periods, and returns this comparison in a matrix.

#### Usage

```r
mcmc.PlotTrackingInc(
  draws,
  T.cal,
  T.tot,
  actual.inc.tracking.data,
  xlab = "Week",
  ylab = "Transactions",
  xticklab = NULL,
  title = "Tracking Weekly Transactions",
  ymax = NULL,
  sample_size = 10000,
  covariates = NULL,
  legend = c("Actual", "Model")
)
```

#### Arguments

- **draws**: MCMC draws as returned by `*.mcmc.DrawParameters`
- **T.cal**: A vector to represent customers’ calibration period lengths (in other words, the `T.cal` column from a customer-by-sufficient-statistic matrix). Considering rounding in order to speed up calculations.
- **T.tot**: End of holdout period. Must be a single value, not a vector.
`mcmc.PlotTrackingInc`

- `actual.inc.tracking.data`  
  A vector containing the incremental number of repeat transactions made by customers for each period in the total time period (both calibration and holdout periods).

- `xlab`  
  Descriptive label for the x axis.

- `ylab`  
  Descriptive label for the y axis.

- `xticklab`  
  A vector containing a label for each tick mark on the x axis.

- `title`  
  Title placed on the top-center of the plot.

- `ymax`  
  Upper boundary for y axis.

- `sample_size`  
  Sample size for estimating the probability distribution. See `mcmc.ExpectedCumulativeTransactions`.

- `covariates`  
  (optional) Matrix of covariates, for Pareto/NBD (Abe) model, passed to `abe.GenerateData` for simulating data.

- `legend`  
  plot legend, defaults to 'Actual' and 'Model'.

**Details**

The expected transactions need to be sampled. Due to this sampling, the return result varies from one call to another. Larger values of `sample_size` will generate more stable results.

**Value**

Matrix containing actual and expected incremental repeat transactions.

**See Also**

- `mcmc.PlotTrackingCum`
- `mcmc.ExpectedCumulativeTransactions`
- `elog2inc`

**Examples**

```r
## Not run:
data("groceryElog")
cbs <- elog2cbs(groceryElog, T.cal = "2006-12-31")
inc <- elog2inc(groceryElog)
param.draws <- pnbd.mcmc.DrawParameters(cbs)
mat <- mcmc.PlotTrackingInc(param.draws,
  T.cal = cbs$T.cal,
  T.tot = max(cbs$T.cal + cbs$T.star),
  actual.inc.tracking.data = inc)

## End(Not run)
```
mcmc.pmf

Probability Mass Function for Pareto/GGG, Pareto/NBD (HB) and Pareto/NBD (Abe)

Description

Return the probability distribution of purchase frequencies for a random customer in a given time period, i.e. \( P(X(t) = x) \). This is estimated by generating \( \text{sample\_size} \) number of random customers that follow the provided parameter draws. Due to this sampling, the return result varies from one call to another.

Usage

mcmc.pmf(draws, t, x, sample_size = 10000, covariates = NULL)

Arguments

draws MCMC draws as returned by \*.mcmc.DrawParameters
t Length of time for which we are calculating the expected number of transactions. May also be a vector.
x Number of transactions for which probability is calculated. May also be a vector.
sample_size Sample size for estimating the probability distribution.
covariates (optional) Matrix of covariates, for Pareto/NBD (Abe) model, passed to \texttt{abe.GenerateData} for simulating data.

Value

\( P(X(t) = x) \). If either \( t \) or \( x \) is a vector, then the output will be a vector as well. If both are vectors, the output will be a matrix.

Examples

data(“groceryElog”)  
cbs <- elog2cbs(groceryElog)  
param.draws <- pnbd.mcmc.DrawParameters(cbs,  
  mcmc = 100, burnin = 50, thin = 10, chains = 1) # short MCMC to run demo fast  
mcmc.pmf(param.draws, t = c(26, 52), x = 0:6)
mcmc.setBurnin

(Re-)set burnin of MCMC chains.

Description

(Re-)set burnin of MCMC chains.

Usage

mcmc.setBurnin(draws, burnin)

Arguments

draws MCMC draws as returned by *.mcmc.DrawParameters
burnin New start index.

Value

2-element list with MCMC draws

Examples

data("groceryElog")
cbs <- elog2cbs(groceryElog)
param.draws <- pnbd.mcmc.DrawParameters(cbs,
    mcmc = 100, burnin = 50, thin = 10, chains = 1) # short MCMC to run demo fast
param.draws.stable <- mcmc.setBurnin(param.draws, burnin = 80)

nbd.cbs.LL

Calculate the log-likelihood of the NBD model

Description

Calculate the log-likelihood of the NBD model

Usage

nbd.cbs.LL(params, cal.cbs)

Arguments

params NBD parameters - a vector with r and alpha, in that order.
cal.cbs Calibration period CBS. It must contain columns for frequency x and total time observed T.cal.
nbd.ConditionalExpectedTransactions

Value

The total log-likelihood for the provided data.

Examples

data("groceryElog")
cbs <- elog2cbs(groceryElog)
params <- nbd.EstimateParameters(cbs)
nbd.cbs.LL(params, cbs)

nbd.ConditionalExpectedTransactions

NBD Conditional Expected Transactions

Description

Uses NBD model parameters and a customer's past transaction behavior to return the number of transactions they are expected to make in a given time period.

Usage

nbd.ConditionalExpectedTransactions(params, T.star, x, T.cal)

Arguments

params NBD parameters - a vector with r and alpha, in that order.
T.star Length of time for which we are calculating the expected number of transactions.
x Number of repeat transactions in the calibration period T.cal, or a vector of calibration period frequencies.
T.cal Length of calibration period, or a vector of calibration period lengths.

Value

Number of transactions a customer is expected to make in a time period of length t, conditional on their past behavior. If any of the input parameters has a length greater than 1, this will be a vector of expected number of transactions.

Examples

data("groceryElog")
cbs <- elog2cbs(groceryElog, T.cal = "2006-12-31")
params <- nbd.EstimateParameters(cbs)
xstar.est <- nbd.ConditionalExpectedTransactions(params, cbs$T.star, cbs$x, cbs$T.cal)
sum(xstar.est) # expected total number of transactions during holdout
**nbd.EstimateParameters**

*Parameter Estimation for the NBD model*

**Description**

Estimates parameters for the NBD model via Maximum Likelihood Estimation.

**Usage**

```r
nbd.EstimateParameters(cal.cbs, par.start = c(1, 1), max.param.value = 10000)
```

**Arguments**

- `cal.cbs`: Calibration period CBS. It must contain columns for frequency `x` and total time observed `T.cal`.
- `par.start`: Initial NBD parameters - a vector with `r` and `alpha` in that order.
- `max.param.value`: Upper bound on parameters.

**Value**

List of estimated parameters.

**References**


**Examples**

```r
data("groceryElog")
cbs <- elog2cbs(groceryElog)
nbd.EstimateParameters(cbs)
```

---

**nbd.GenerateData**

*Simulate data according to NBD model assumptions*

**Description**

Simulate data according to NBD model assumptions.

**Usage**

```r
nbd.GenerateData(n, T.cal, T.star, params, date.zero = "2000-01-01")
```
Arguments

- `n` Number of customers.
- `T.cal` Length of calibration period.
- `T.star` Length of holdout period. This may be a vector.
- `params` NBD parameters - a vector with `r` and `alpha` in that order.
- `date.zero` Initial date for cohort start. Can be of class character, Date or POSIXt.

Value

List of length 2:

- `cbs` A data.frame with a row for each customer and the summary statistic as columns.
- `elog` A data.frame with a row for each transaction, and columns `cust`, `date` and `t`.

Examples

```r
n <- 200  # no. of customers
T.cal <- 32  # length of calibration period
T.star <- 32  # length of hold-out period
params <- c(r = 0.85, alpha = 4.45)  # purchase frequency lambda_i ~ Gamma(r, alpha)
data <- nbd.GenerateData(n, T.cal, T.star, params)
cbs <- data$cbs  # customer by sufficient summary statistic - one row per customer
elog <- data$elog  # Event log - one row per event/purchase
```

---

**nbd.LL**

*Calculate the log-likelihood of the NBD model*

**Description**

Calculate the log-likelihood of the NBD model

**Usage**

```
nbd.LL(params, x, T.cal)
```

**Arguments**

- `params` NBD parameters - a vector with `r` and `alpha`, in that order.
- `x` Frequency, i.e. number of re-purchases.
- `T.cal` Total time of observation period.

**Value**

A numeric vector of log-likelihoods.

**See Also**

```
nbd.cbs.LL
```
Simulate data according to Pareto/GGG model assumptions

Usage

```r
pggg.GenerateData(n, T.cal, T.star, params, date.zero = "2000-01-01")
```

Arguments

- `n`: Number of customers.
- `T.cal`: Length of calibration period. If a vector is provided, then it is assumed that customers have different 'birth' dates, i.e. $\max(T.cal) - T.cal$.
- `T.star`: Length of holdout period. This may be a vector.
- `params`: A list of model parameters $r$, $alpha$, $s$, $beta$, $t$ and $gamma$.
- `date.zero`: Initial date for cohort start. Can be of class character, Date or POSIXt.

Value

List of length 2:

- `cbs`: A data.frame with a row for each customer and the summary statistic as columns.
- `elog`: A data.frame with a row for each transaction, and columns cust, date and $t$.

References


Examples

```r
params <- list(t = 4.5, gamma = 1.5, r = 5, alpha = 10, s = 0.8, beta = 12)
data <- pggg.GenerateData(n = 200, T.cal = 32, T.star = 32, params)
cbs <- data$cbs # customer by sufficient summary statistic - one row per customer
elog <- data$elog # Event log - one row per event/purchase
```
Description

Returns draws from the posterior distributions of the Pareto/GGG parameters, on cohort as well as on customer level.

Usage

```r
pggg.mcmc.DrawParameters(
  cal.cbs,
  mcmc = 2500,
  burnin = 500,
  thin = 50,
  chains = 2,
  mc.cores = NULL,
  param_init = NULL,
  trace = 100
)
```

Arguments

- `cal.cbs`: Calibration period customer-by-sufficient-statistic (CBS) data.frame. It must contain a row for each customer, and columns `x` for frequency, `t.x` for recency, `T.cal` for the total time observed, as well as the sum over logarithmic inter-transaction times `litt`. A correct format can be easily generated based on the complete event log of a customer cohort with `elog2cbs`.
- `mcmc`: Number of MCMC steps.
- `burnin`: Number of initial MCMC steps which are discarded.
- `thin`: Only every `thin`-th MCMC step will be returned.
- `chains`: Number of MCMC chains to be run.
- `mc.cores`: Number of cores to use in parallel (Unix only). Defaults to `min(chains, detectCores())`.
- `param_init`: List of start values for cohort-level parameters.

Details

See `demo('pareto-ggg')` for how to apply this model.
Value

List of length 2:

level_1 list of mcmc.list, one for each customer, with draws for customer-level parameters \( k, \lambda, \tau, z, \mu \)

level_2 mcmc.list, with draws for cohort-level parameters \( r, \alpha, s, \beta, t, \gamma \)

References


See Also

pggg.GenerateData mcmc.PAlive mcmc.DrawFutureTransactions

Examples

data("groceryElog")
cbs <- elog2cbs(groceryElog, T.cal = "2006-12-31")
param.draws <- pggg.mcmc.DrawParameters(cbs,
    mcmc = 20, burnin = 10, thin = 2, chains = 1) # short MCMC to run demo fast

# cohort-level parameter draws
as.matrix(param.draws$level_2)
# customer-level parameter draws for customer with ID '4'
as.matrix(param.draws$level_1[['4']])

# estimate future transactions
xstar.draws <- mcmc.DrawFutureTransactions(cbs, param.draws, cbs$T.star)
xstar.est <- apply(xstar.draws, 2, mean)
head(xstar.est)
Arguments

- **draws**: MCMC draws as returned by `pggg.mcmc.DrawParameters`.
- **xmax**: Upper bound for x-scale.
- **fn**: Optional function to summarize individual-level draws for k, e.g. 'mean'.
- **title**: Plot title.

References


Examples

```r
data("groceryElog")
cbs <- elog2cbs(groceryElog, T.cal = "2006-12-31")
param.draws <- pggg.mcmc.DrawParameters(cbs,
  mcmc = 20, burnin = 10, thin = 2, chains = 1) # short MCMC to run demo fast
pggg.plotRegularityRateHeterogeneity(param.draws)
```

Description

Plot timing patterns of sampled customers

Usage

```r
plotTimingPatterns(elog, n = 40, T.cal = NULL, T.tot = NULL, title = "Sampled Timing Patterns", headers = NULL)
```

Arguments

- **elog**: Event log, a data.frame with columns cust and transaction time t or date.
- **n**: Number of sampled customers.
- **T.cal**: End of calibration period, which is visualized as a vertical line.
- **T.tot**: End of observation period
- **title**: Plot title.
- **headers**: Vector of length 2 for adding headers to the plot, e.g. c("Calibration", "Holdout").
pnbd.GenerateData  

Simulate data according to Pareto/NBD model assumptions

Description

Simulate data according to Pareto/NBD model assumptions

Usage

pnbd.GenerateData(n, T.cal, T.star, params, date.zero = "2000-01-01")

Arguments

n  
Number of customers.

T.cal  
Length of calibration period. If a vector is provided, then it is assumed that customers have different 'birth' dates, i.e. \( \max(T.cal) - T.cal \).

T.star  
Length of holdout period. This may be a vector.

params  
A list of model parameters \( r, \alpha, s, \beta \).

date.zero  
Initial date for cohort start. Can be of class character, Date or POSIXt.

Value

List of length 2:

cbs  
A data.frame with a row for each customer and the summary statistic as columns.

elog  
A data.frame with a row for each transaction, and columns cust, date and t.

Examples

params <- list(r = 5, alpha = 10, s = 0.8, beta = 12)
data <- pnbd.GenerateData(n = 200, T.cal = 32, T.star = 32, params)
cbs <- data$cbs # customer by sufficient summary statistic - one row per customer
elog <- data$elog  # Event log - one row per event/purchase
pnbd.mcmc.DrawParameters

Pareto/NBD (HB) Parameter Draws

Description

Returns draws from the posterior distributions of the Pareto/NBD (HB) parameters, on cohort as well as on customer level.

Usage

pnbd.mcmc.DrawParameters(
  cal.cbs,
  mcmc = 2500,
  burnin = 500,
  thin = 50,
  chains = 2,
  mc.cores = NULL,
  use_data_augmentation = TRUE,
  param_init = NULL,
  trace = 100
)

Arguments

cal.cbs Calibration period customer-by-sufficient-statistic (CBS) data.frame. It must contain a row for each customer, and columns x for frequency, t.x for recency and T.cal for the total time observed. A correct format can be easily generated based on the complete event log of a customer cohort with elog2cbs.

mcmc Number of MCMC steps.

burnin Number of initial MCMC steps which are discarded.

thin Only every thin-th MCMC step will be returned.

chains Number of MCMC chains to be run.

mc.cores Number of cores to use in parallel (Unix only). Defaults to \text{min(chains, detectCores())}.

use_data_augmentation deprecated

param_init List of start values for cohort-level parameters.

trace Print logging statement every trace-th iteration. Not available for mc.cores > 1.

Details

See demo('pareto-ggg') for how to apply this model.
pnbd.mcmc.DrawParameters

Value

2-element list:

- \texttt{level\_1} list of \texttt{mcmc.list}, one for each customer, with draws for customer-level parameters \( \lambda, \tau, z, \mu \)
- \texttt{level\_2} \texttt{mcmc.list}, with draws for cohort-level parameters \( r, \alpha, s, \beta \)

References


See Also

\texttt{pnbd.GenerateData mcmc.DrawFutureTransactions mcmc.PAlive}

Examples

data("groceryElog")
cbs <- elog2cbs(groceryElog, T.cal = "2006-12-31")
param.draws <- pnbd.mcmc.DrawParameters(cbs,
    mcmc = 100, burnin = 50, thin = 10, chains = 1) # short MCMC to run demo fast

# cohort-level parameter draws
as.matrix(param.draws$level\_2)
# customer-level parameter draws for customer with ID '4'
as.matrix(param.draws$level\_1[["4"]])

# estimate future transactions
xstar.draws <- mcmc.DrawFutureTransactions(cbs, param.draws, cbs$T.star)
xstar.est <- apply(xstar.draws, 2, mean)
head(xstar.est)
Index

* datasets
  groceryElog, 9
  abe.GenerateData, 3, 5, 28, 33, 35, 36
  abe.mcmc.DrawParameters, 4
  bgcnbd.cbs.LL (mbgcnbd.cbs.LL), 10
  bgcnbd.ConditionalExpectedTransactions
    (mbgcnbd.ConditionalExpectedTransactions), 11
  bgcnbd.EstimateParameters
    (mbgcnbd.EstimateParameters), 12
  bgcnbd.Expectation
    (mbgcnbd.Expectation), 13
  bgcnbd.ExpectedCumulativeTransactions
    (mbgcnbd.ExpectedCumulativeTransactions), 14
  bgcnbd.GenerateData
    (mbgcnbd.GenerateData), 15
  bgcnbd.LL (mbgcnbd.cbs.LL), 10
  bgcnbd.PAlive (mbgcnbd.PAlive), 16
  bgcnbd.PlotFrequencyInCalibration
    (mbgcnbd.PlotFrequencyInCalibration), 17
  bgcnbd.PlotFreqVsConditionalExpectedFrequency
    (mbgcnbd.PlotFreqVsConditionalExpectedFrequency), 19
  bgcnbd.PlotRecVsConditionalExpectedFrequency
    (mbgcnbd.PlotRecVsConditionalExpectedFrequency), 20
  bgcnbd.PlotTrackingCum
    (mbgcnbd.PlotTrackingCum), 21
  bgcnbd.PlotTrackingInc
    (mbgcnbd.PlotTrackingInc), 23
  bgcnbd.pmf (mbgcnbd.pmf), 25
  mbgnbd.EstimateParameters, 13
  difftime, 6

elog2cbs, 4, 5, 10, 12, 42, 46
elog2cum, 7, 33
elog2inc, 35
elog2inc (elog2cum), 7
estimateRegularity, 8
groceryElog, 9
mbgcnbd.cbs.LL, 10
mbgcnbd.ConditionalExpectedTransactions, 11
mbgcnbd.EstimateParameters, 12
mbgcnbd.Expectation, 13
mbgcnbd.ExpectedCumulativeTransactions, 14, 23, 24
mbgcnbd.GenerateData, 15
mbgcnbd.LL (mbgcnbd.cbs.LL), 10
mbgcnbd.PAlive, 16
mbgcnbd.PlotFrequencyInCalibration, 17
mbgcnbd.PlotFreqVsConditionalExpectedFrequency, 19
mbgcnbd.PlotRecVsConditionalExpectedFrequency, 20
mbgcnbd.PlotTrackingCum, 21
mbgcnbd.PlotTrackingInc, 23
mbgcnbd.pmf, 25
mbgcnbd.EstimateParameters
  (mbgcnbd.EstimateParameters), 12
mcmc.DrawFutureTransactions, 5, 26, 29, 32, 43, 47
mcmc.Expectation, 27
mcmc.ExpectedCumulativeTransactions, 28, 33, 35
mcmc.list, 5, 43, 47
mcmc.PActive, 29
mcmc.PAlive, 5, 30, 43, 47
mcmc.PlotFrequencyInCalibration, 30
mcmc.plotPActiveDiagnostic, 32
mcmc.PlotTrackingCum, 32, 35
mcmc.PlotTrackingInc, 33, 34
mcmc.pmf, 31, 36
mcmc.setBurnin, 37

nbd.cbs.LL, 37, 40
nbd.ConditionalExpectedTransactions, 38
nbd.EstimateParameters, 39
nbd.GenerateData, 39
nbd.LL, 40

pggg.GenerateData, 41, 43
pggg.mcmc.DrawParameters, 42, 44
pggg.plotRegularityRateHeterogeneity, 43
plotTimingPatterns, 44
pnbd.GenerateData, 45, 47
pnbd.mcmc.DrawParameters, 46