Biblioteca `IBGEPesq`
Manual

Eloane G. Ramos
Coordenação de Trabalho e Rendimento

Giuseppe A. Antonaci
Coordenação de Metodologia e Qualidade

IBGE

16 de fevereiro de 2007

Sumário

1 Introdução

O pacote `IBGEPesq_1.0-1.zip` contém algumas funções úteis para leitura e manipulação dos dados da PNAD:

- `le.pesquisa()`. Permite a leitura dos microdados da PNAD ou da PME;
- `merge.blocos()`. Realiza um merge em blocos, permitindo a manipulação de grande volume de dados, como é o caso da PNAD;
- `media.pond()`. Calcula médias ponderadas por domínios;
- `IBGEtabula()`. Estima totais e CVs, utilizando o pacote survey, permitindo a desagregação em múltiplos domínios.

Além destas funções o pacote também possui alguns arquivos de dados:

- `dicPME`. Dicionário de variáveis da Pesquisa Mensal de Emprego do IBGE;
- `dicdom2005`. Dicionário de domicílios da Pesquisa Nacional por Amostragem Domiciliar do IBGE de 2005;
- `dicpes2005`. Dicionário de pessoas da Pesquisa Nacional por Amostragem Domiciliar do IBGE de 2005;
- `ocupacao2005`. Dados referentes às condições de ocupação da PNAD 2005, utilizado para os exemplos do pacote.

O arquivo de microdados de pessoas da PNAD 2005 contêm 408148 registros e mais de 300 variáveis, possuindo um tamanho de mais 300MB.

Para superar as dificuldades do R em lidar com grandes volumes de dados, a função `le.pesquisa()` implementa uma leitura em blocos, assumindo que o usuário solicitará apenas as variáveis que vai efetivamente utilizar. Como exemplo, para tabular dados da PNAD sobre educação, necessita-se apenas das variáveis referentes a esse tema (anos de estudo, rede de ensino, ...), além de outras variáveis mais gerais como idade, sexo e peso amostral. Num Pentium IV com 2.8GHz e 512MB de memória conseguiu-se ler e processar sem problemas até 20 variáveis da PNAD.

O tempo de leitura depende tanto da quantidade de variáveis que serão lidas, quanto do desempenho do computador utilizado. Num Pentinum IV de 2.8GHz e memória RAM de **1GB**, a leitura de 28 variáveis dos microdados de pessoas de 2005 levou aproximadamente 10 minutos.

Para ler e trabalhar com mais de 20 variáveis de pessoas da PNAD ao mesmo tempo, recomenda-se que o computador possua **1GB de memória RAM** ou mais.

2 Função `le.pesquisa()`

Realiza a leitura dos microdados da PNAD, em blocos, recuperando apenas as variáveis que se deseja utilizar num objeto do tipo data-frame.

Os microdados são arquivos do tipo texto, onde as colunas de dados são dispostas lado a lado sem nenhuma separação. Na realidade, qualquer arquivo de dados do tipo texto que esteja nesse mesmo formato pode ser lido pela função `le.pesquisa()`. Basta que se forneça um dicionário R do arquivo de dados desejado.

Os dicionários R da PNAD são fornecidos juntamente com essa biblioteca e podem ser carregados usando os comandos `data(dicpes2005)` e `data(dicdom2005)`. O primeiro se refere aos microdados de pessoas e o segundo aos microdados de domicílios. Ambos possuem as seguintes colunas:

- cod. Contém os códigos das variáveis da pesquisa (Ex: V8005, V0201, ...).
- inicio. Contém as posições iniciais de cada variável dentro do arquivo de microdados.
- tamanho. Contém o tamanho total de cada variável dentro do arquivo de dados.
- desc. Contém uma pequena descrição de cada variável da PNAD.

Não é necessário que o usuário conheça as posições e tamanhos das variáveis que deseja usar. Os dicionários R `dicpes2005` e `dicdom2005` servem justamente para que o usuário possa especificar as variáveis apenas pelo código.

2.1 Sintaxe da função `le.pesquisa()`

```
le.pesquisa(dicionario, pathname.in, pathname.out=NA, codigos, nomes=NA, quant=NA,
            tbloco=2000)
```

Argumentos de entrada

dicionario Variável obrigatória do tipo data-frame contendo o dicionário R dos microdados que serão lidos: `dicpes2005` para os microdados de pessoas e `dicdom2005` para os microdados de domicílios. Estes objetos são fornecidos pelo IBGE juntamente com este pacote.

pathname.in Variável obrigatória do tipo caracter com o caminho e nome do arquivo de microdados a ser lido.

pathname.out Variável opcional do tipo caracter com o caminho e o nome do arquivo de saída onde serão armazenados os dados lidos (geralmente no formato *.Rdata). Se esse argumento não for especificado, nenhum arquivo será salvo.

codigos Variável obrigatória, do tipo vetor de caracteres, com os códigos das variáveis que se deseja ler, conforme especificado no dicionário da pesquisa.

nomes Variável opcional, do tipo vetor de caracteres, com os nomes que se deseja dar às variáveis lidas. Caso não seja especificado, os nomes das colunas do data-frame resultante serão iguais ao argumento `codigos`.

quant Variável opcional, do tipo inteiro, com a quantidade de linhas contidas no arquivo de microdados que se deseja ler. É utitilizada apenas para a construção da barra de espera que indica o andamento da leitura em valores percentuais. Em 2005, os microdados de pessoas da PNAD contêm 408148 linhas e o de domicílios 142471 linhas.

tbloco Variável opcional, do tipo inteiro, com o tamanho do bloco de leitura. Se não for especificada, serão lidas 2000 linhas por vez, valor adequado num computador com memória RAM de 512MB. Para um computador com menos de 512MB de memória, `tbloco` deve ser menor que 2000. Para um computador com mais de 512MB de memória, `tbloco` pode ser maior que 2000, o que reduzirá o tempo de leitura.

Argumento de saída

Como resultado da leitura é fornecido um objeto do tipo data-frame apenas com as variáveis escolhidas e com o número de linhas igual ao do arquivo original de microdados. Os nomes das colunas do data-frame de saída são os contidos no argumento `nomes`. Caso este não tenha sido especificado, são utilizados os elementos do argumento `codigos`.

2.2 Exemplo da função `le.pesquisa()`

Deseja-se ler as seguintes variáveis do arquivo de pessoas:

código	descrição	novo nome
V4703	anos de estudo	ANOSEST
V4704	condição de atividade	CONDATV
V4705	condição de ocupação	CONDOCP
V4707	horas trabalhadas	HTRAB
V4729	peso amostral da pessoa	PESO

e armazenar no arquivo `dados.Rdata`. A rotina a seguir realiza a leitura, assumindo que os microdados `PES2005.TXT` estão na pasta `E:/DADOS/`:

```
library(IBGEPesq)
data(dicpes2005)
caminho.microdados <- file.path("E:","DADOS","PES2005.TXT")
dados <- le.pesquisa(dicionario=dicpes2005, pathname.in=caminho.microdados,
                     pathname.out="dados.Rdata",
```

```
codigos=c("V4703","V4704","V4705","V4707","V4729"),
nomes=c("ANOSEST","CONDATV","CONDOCP","HTRAB","PESO"),
quant=408148)
```

Ao final da execução, os dados lidos estarão no data-frame `dados`, além de terem sido gravados no arquivo `dados.Rdata` para uso posterior.

3 Função `merge.blocos()`

Realiza um *merge* em blocos, permitindo a manipulação de grandes volumes de dados, como é o caso da PNAD.

3.1 Sintaxe da função `merge.blocos()`

```
merge.blocos(dados1, dados2, tbloco=3000)
```

Argumentos de entrada

dados1, dados2 Variáveis obrigatórias, do tipo matriz ou data-frame, que serão incorporadas em um só objeto, utilizando como colunas chaves aquelas cujos nomes são comuns entre os objetos `dados1` e `dados2`.

tbloco Variável opcional, que especifica o número de linhas que serão processadas de cada vez. Se não for especificado, assume-se o valor 3000, que foi utilizado eficientemente num computador com 512MB de memória RAM. Para computadores com menos memória, `tbloco` deve ser menor que 3000.

Argumento de saída

É retornado um novo data-frame, consistindo da incorporação dos objetos `dados1` e `dados2`. O novo objeto conterá um número de colunas igual a soma dos números de colunas de `dados1` e `dados2`, menos o número de colunas cujos nomes são comuns entre os dois objetos.

3.2 Exemplo da função `merge.blocos()`

Deseja-se ler as seguintes variáveis dos microdados de pessoas e de domicílios, mesclá-las e gravá-las no arquivo `dadosfamilias.Rdata`:

Dados a serem lidos dos microdados de pessoas:

código	descrição	novo nome
UF	Unidade da Federação	UF
V0102	Código de controle	CONTROLE
V0103	Série	SERIE
V0302	Sexo	SEXO
V0402	Condição na família	COND_FAMILIA
V4704	Condição de atividade	COND_ATIVIDADE
V4729	Peso amostral	PESO

Dados a serem lidos dos microdados de domicílios:

código	descrição	novo nome
V0102	Código de controle	CONTROLE
V0103	Série	SERIE
V0201	Espécie do domicílio (particular improvisado, permanente, ...)	ESPECIE_DOM

A rotina a seguir realiza a leitura, assumindo que os microdados PES2005.TXT e DOM2005.TXT estão na pasta E:/DADOS/:

```
library(IBGEPesq)

### leitura dos dados de pessoas
data(dicpes2005)
nomesdic <- c("UF", "V0102", "V0103", "V0302", "V0402", "V4704", "V4729")
nomes <- c("UF", "CONTROLE", "SERIE", "SEXO", "COND_FAMILIA", "COND_ATIVIDADE",
           "PESOS")
caminho.microdados <- file.path("E:", "DADOS", "PES2005.TXT")
nreg = 408148
dadospes <- le.pesquisa(dicionario=dicpes2005, pathname.in=caminho.microdados,
                        codigos=nomesdic, nomes=nomes, quant=nreg)

### leitura dos dados de domicilios
data(dicdom2005)
nomesdic <- c("V0102", "V0103", "V0201")
nomes <- c("CONTROLE", "SERIE", "ESPECIE_DOM")
caminho.microdados <- file.path("E:", "DADOS", "DOM2005.TXT")
nreg = 142471
dadosdom <- le.pesquisa(dicionario=dicdom2005, pathname.in=caminho.microdados,
                        codigos=nomesdic, nomes=nomes, quant=nreg)

### cruzamento dos dados de domicílios com os de pessoas
dados <- merge.blocos(dadospes, dadosdom)
rm(dadospes, dadosdom)
arq.saida <- "dadosfamilias.Rdata"
save(dados, file=arq.saida)
```

4 Função media.pond()

Calcula média ponderada por domínios.

4.1 Sintaxe da função media.pond()

```
media.pond(x, pesos, dom)
```

Será calculada a média da variável x, ponderada pela variável pesos, para cada domínio existente na variável dom. Múltiplos domínios são permitidos.

Argumentos de entrada

x Variável obrigatória, do tipo vetor, matriz ou data-frame.

pesos Variável obrigatória, do tipo vetor numérico, que especifica os pesos segundo os quais a média será ponderada.

dom Variável obrigatória, do tipo vetor ou lista, que especifica os domínios para os quais serão calculadas médias ponderadas da variável x.

Se x for matriz ou data-frame, dom deve ser um vetor com número de elementos igual ao número de linhas de x. Nesse caso, será calculada uma média ponderada para cada coluna de x e para cada domínio contido na variável dom.

Se x for vetor, dom pode ser vetor ou uma lista com vários vetores do mesmo tamanho que x. Nesse caso, serão calculadas médias cruzadas da variável x entre os vários níveis de domínios existentes em dom.

Argumento de saída

Se x for matriz ou data-frame, o resultado é uma matriz com mesmo número de colunas que x e com número de linhas igual a quantidade de domínios definidos no vetor dom.

Se ambos x e dom forem vetores, o resultado é um vetor com número de elementos igual a quantidade de domínios definidos no vetor dom.

Se x for vetor e dom for uma lista de vetores, o resultado é uma matriz ou um array onde a primeira dimensão é igual à quantidade de domínios definidos no primeiro vetor de dom, a segunda dimensão é igual à quantidade de domínios definidos no segundo vetor de dom e, assim por diante.

4.2 Exemplo da função media.pond()

Deseja-se calcular:

1. Percentual de pessoas que possuem internet e média de anos de estudo por grupos de rendimento.
2. Percentual de pessoas que possuem internet por grupos de rendimento e por grupos de idade.

A rotina a seguir calcula as médias desejadas para grupos arbitrários de rendimento e de idade:

```
library(IBGEPesq)

### criação de dados fictícios:
n <- 1000
internet <- sample(c(T,F), n, replace=T) #variável dicotômica
anos.estudo <- sample(0:15, n, replace=T) #variável numérica
x = data.frame(internet, anos.estudo)
rendimento <- sample(0:999, n, replace=T)
idade <- sample(1:74, n, replace=T)
pesos <- sample(seq(100, 1100, 50), n, replace=T)

### criação de grupos arbitrários de idade e rendimento:
gruposrend <- trunc(rendimento/200)
gruposrend <- factor(gruposrend, labels= c("Menor que R$200", "R$200 a R$399",
                    "R$400 a R$599", "R$600 a R$799", "R$800 a R$999"))
gruposidade <- trunc(idade/15)
gruposidade <- factor(gruposidade, labels= c("Menor que 15", "de 15 a 29",
                    "de 30 a 44", "de 45 a 59", "60 ou mais"))
### Percentual de pessoas que possuem internet e média de anos de estudo
### por grupos de rendimento:
m.rend <- media.pond(x=x, pesos=pesos, dom=gruposrend)
m.rend[,1] <- m.rend[,1]*100

### Percentual de pessoas que possuem internet por grupos de rendimento e
### por grupos de idade:
m.rend.idade <- media.pond(x=x$internet,pesos=pesos,
                           dom=list(gruposrend,gruposidade))*100
```

5 Função IBGEtabula()

Estima totais e CVs, utilizando o pacote survey, permitindo a desagregação em múltiplos domínios.

5.1 Sintaxe da função IBGEtabula()

```
estimativa <- IBGEtabula(desenho, colunas, dominios, elimina=NA,
                         dominios.vazios=T, total.geral=F)
```

Constrói uma tabela de totais e proporções com seus respectivos coeficientes de variação considerando o desenho amostral complexo da amostra. Múltiplos domínios são permitidos.

Argumentos de entrada

desenho Objeto de classe `survey.design` obtido pela função svydesign do pacote `survey`. Contém as informações do desenho amostral e os dados a serem tabulados.

colunas Objeto do tipo vetor de characteres com os nomes das colunas dos dados contidos em `desenho` para as quais se deseja estimar totais. As variáveis indicadas nesta coluna devem ser do tipo binário (0 ou 1).

dominios Objeto do tipo vetor de characteres com os nomes das colunas dos dados contidos em `desenho` que definem os domínios de estimação, em ordem de precedência. Veja mais detalhes abaixo.

elimina Objeto do tipo lista, com um elemento para cada domínio definido no argumento `dominios`. Cada elemento da lista é um vetor de caracteres, com itens que se deseja eliminar da tabela final como, por exemplo, sem declaração, ou NA, caso nenhum elemento do domínio deva ser eliminado.Se for fornecido somente um elemento este será utilizado em todos os domínios.

dominios.vazios Objeto do tipo lógico (boolean) que determina se cruzamentos de domínios vazios devem ser mantidos na tabela ou eliminados.

digitos.cv Número de dígitos depois da vírgula para os valores de CV em porcentagem.

total.geral Objeto do tipo lógico (boolean) que determina se deve ser calculada uma linha de total geral.

Os dados a serem tabulados são definidos dentro do argumento `desenho`, que deve conter também a estrutura do desenho amostral utilizado. Este objeto, de classe `survey.design`, pode ser obtido utilizando-se a função `svydesign`, disponível no pacote `survey`.

Serão calculados totais e CVs para cada variável definida em `colunas`, desagregados pelos domínios definidos em `dominios`. As colunas especificadas por esses dois argumentos devem existir nos dados contidos em `desenho`.

IMPORTANTE: As colunas de domínio devem ser do tipo fator. Os `labels` desses fatores serão usados para nomear as linhas da tabela final. Os 'labels' não devem conter o sinal de ponto (.) nos nomes.

Itens indesejados de cada domínio devem ser especificados no argumento `elimina`. Este argumento deve ser do tipo lista, com número de elementos igual ao número de domínios.

Argumento de saída

Um objeto do tipo lista com dois elementos do tipo matriz: `totais` e `cv`. O número de colunas de cada matriz é igual ao tamanho do argumento `colunas`. As linhas são formadas pelos vários cruzamentos entre os `domínios`, formatadas e identadas de acordo com a hierarquia entre eles.

5.2 Exemplo da função IBGEtabula()

Deseja-se construir uma tabela com o número de pessoas separadas nos domínios: Condição na ocupação, Sexo, Grupos de idade e Região geográfica.

```
library(survey)
library(IBGEPesq)
data(ocupacao2005)

colnames(dados) <- c("UF", "IDADE", "SEXO", "ANOS_ESTUDO", "COND_ATIVIDADE",
 "COND_OCUPACAO", "PREVIDENCIA", "SINDICATO", "ATIVIDADE", "OCUPACAO", "PESOS",
 "gr.anos", "ESP_DOM", "NUM_MOR", "PROJECAO", "INV_FRA", "STRAT", "PSU", "SUBA")

dados <- dados[,c("UF", "SEXO", "COND_OCUPACAO", "PESOS", "gr.anos",
               "PROJECAO", "INV_FRA", "STRAT", "PSU", "SUBA")]
dados <- transform(dados,
pestrato = PROJECAO,
pesos = INV_FRA * SUBA,
brasil = rep(1,nrow(dados)),
  norte = (trunc(UF/10) %in% 1)*1,
  nordeste = (trunc(UF/10) %in% 2)*1,
  sudeste = (trunc(UF/10) %in% 3)*1,
  sul = (trunc(UF/10) %in% 4)*1,
  centroeste = (trunc(UF/10) %in% 5)*1)

dados <- transform(dados, pestrato=factor(pestrato),
                   gr.anos=factor(gr.anos,labels=c("Menos de 1 ano",
                  "1 a 3 anos", "4 a 7 anos", "8 a 10 anos",
                  "11 anos ou mais", "SD")),
                  SEXO = factor(SEXO, labels=c("Homens", "Mulheres")),
                  COND_OCUPACAO = factor(COND_OCUPACAO, labels=c("Ocupados",
                  "Desocupados", "SD"), exclude=NULL))

levels(dados$pestrato)<-1:length(levels(dados$pestrato))
pos.estratos <- data.frame(pestrato=levels(dados$pestrato),
Freq=sort(unique(dados$PROJECAO)))

options(survey.lonely.psu="average", survey.ultimate.cluster=T)
dpnad<-svydesign(ids=~PSU, strata=~STRAT, data=dados, nest=TRUE,
weights=~pesos)
dpnad<-postStratify(dpnad, ~pestrato, pos.estratos) #pós - estratificação

colunas <- c("brasil","norte","nordeste","sudeste","sul","centroeste")
grupos <- c("COND_OCUPACAO","SEXO","gr.anos")
elimina <- list("SD", NA, "SD")

estimativa <- IBGEtabula(dpnad, colunas, grupos, elimina, total.geral=T)
```