Package 'pan'

February 10, 2016

Version 1.4

Date 2016-02-10

Title Multiple Imputation for Multivariate Panel or Clustered Data

Author Original by Joseph L. Schafer

Maintainer Jing hua Zhao <jinghua.zhao@mrc-epid.cam.ac.uk>

Suggests mitools, lme4

LazyData Yes

LazyLoad Yes

Description Multiple imputation for multivariate panel or clustered data.

License GPL-3

NeedsCompilation yes

License_restricts_use no

Repository CRAN

Date/Publication 2016-02-10 18:57:27

R topics documented:

bitest 2
ecme 3
marijuana 6
pan 7
pan.bd 12
YWC.data 13

Index **15**

bitest *A testing example for bivariate imputation*

Description

From: Maria Valaste [mailto:maria.valaste@helsinki.fi]
Sent: 10 December 2012 18:45
To: Jing Hua Zhao
Subject: RE: correction on previous email: Help for using R package PAN

Dear Jing hua Zhao,

Thank You very much! This was just what I intended.
Of course you can include the example in the documentation. It's very good idea.

Best regards,
Maria Valaste

Lainaus "Jing Hua Zhao" <JingHua.Zhao@mrc-epid.cam.ac.uk>:

> Dear Maria,
>
> I just had time to try out your data/code; is this close to what you intend?
>
...
>
> If you don't mind I could include your example in the documentation to
> let others share your experiment.
>
> Best wishes,
>
>
> Jing Hua
>

From: Maria Valaste
[mailto:maria.valaste@helsinki.fi]
Sent: 08 December 2012 19:09
To: Jing Hua Zhao
Subject: Help for using R package PAN

Dear Jing hua Zhao,

I'm trying to use R package PAN. For one Y variable I have managed succesfully to impute using package PAN. When I try to impute two Y variables, I have encountered a

problem that I can't solve. Could please help me? I'm not expert on R program and I'm really desperate.

I have R version 2.15.2 (Platform: x86_64-w64-mingw32/x64 (64-bit)). Below is code that I have tried to run for two Y variables. There also the error message (Error: subscript out of bounds). I also attach the test data below after the R code.

Best regards,
Maria Valaste
University of Helsinki, Finland

Usage

```
data(bitest)
```

Format

A data frame

Source

Maria Valaste <maria.valaste@helsinki.fi>

---

ecme *ECME algorithm for general linear mixed model*

Description

Performs maximum-likelihood estimation for generalized linear mixed models. The model, which is typically applied to longitudinal or clustered responses, is

yi = Xi%*%beta + Zi%*%bi + ei , i=1,. . . ,m,

where

yi = (ni x 1) response vector for subject or cluster i;

Xi = (ni x p) matrix of covariates;

Zi = (ni x q) matrix of covariates;

beta = (p x 1) vector of coefficients common to the population (fixed effects);

bi = (q x 1) vector of coefficients specific to subject or cluster i (random effects); and

ei = (ni x 1) vector of residual errors.

The vector bi is assumed to be normally distributed with mean zero and unstructured covariance matrix psi,

bi $\sim$ N(0,psi) independently for i=1,. . . ,m.

The residual vector ei is assumed to be

ei $\sim$ N(0,sigma2*Vi)

where Vi is a known (ni x ni) matrix. In most applications, Vi is the identity matrix.

Usage

```
ecme(y, subj, occ, pred, xcol, zcol=NULL, vmax, start,
     maxits=1000, eps=0.0001, random.effects=F)
```

Arguments

y	vector of responses. This is simply the individual yi vectors stacked upon one another. Each element of y represents the observed response for a particular subject-occasion, or for a particular unit within a cluster.
subj	vector of same length as y, giving the subject (or cluster) indicators i for the elements of y. For example, suppose that y is in fact c(y1,y2,y3,y4) where length(y1)=2, length(y2)=3, length(y3)=2, and length(y4)=7. Then subj should be c(1,1,2,2,2,3,3,4,4,4,4,4,4,4).
occ	vector of same length as y indicating the "occasions" for the elements of y. In a longitudinal dataset where each individual is measured on at most nmax distinct occasions, each element of y corresponds to one subject-occasion, and the elements of of occ should be coded as 1,2,...,nmax to indicate these occasion labels. (You should label the occasions as 1,2,...,nmax even if they are not equally spaced in time; the actual times of measurement will be incorporated into the matrix "pred" below.) In a clustered dataset, the elements of occ label the units within each cluster i, using the labels 1,2,...,ni.
pred	matrix of covariates used to predict y. The number of rows should be length(y). The first column will typically be constant (one), and the remaining columns correspond to other variables appearing in Xi and Zi.
xcol	vector of integers indicating which columns of pred will be used in Xi. That is, pred[,xcol] is the Xi matrices (stacked upon one another.
zcol	vector of integers indicating which columns of pred will be used in Zi. That is, pred[,zcol] is the Zi matrices (stacked upon one another). If zcol=NULL then the model is assumed to have no random effects; in that case the parameters are estimated noniteratively by generalized least squares.
vmax	optional matrix of dimension c(max(occ),max(occ)) from which the Vi matrices will be extracted. In a longitudinal dataset, vmax would represent the Vi matrix for an individual with responses at all possible occasions 1,2,...,nmax=max(occ); for individuals with responses at only a subset of these occasions, the Vi will be obtained by extracting the rows and columns of vmax for those occasions. If no vmax is specified by the user, an identity matrix is used. In most applications of this model one will want to have Vi = identity, so most of the time this argument can be omitted.
start	optional starting values of the parameters. If this argument is not given then ecme() chooses its own starting values. This argument should be a list of three elements named "beta", "psi", and "sigma2". Note that "beta" should be a vector of the same length as "xcol", "psi" should be a matrix of dimension c(length(zcol),length(zcol)), and "sigma2" should be a scalar. This argument has no effect if zcol=NULL.

maxits	maximum number of cycles of ECME to be performed. The algorithm runs to convergence or until "maxits" iterations, whichever comes first.
eps	convergence criterion. The algorithm is considered to have converged if the relative differences in all parameters from one iteration to the next are less than eps–that is, if all(abs(new-old)<eps*abs(old)).
random.effects	if TRUE, returns empirical Bayes estimates of all the random effects bi (i=1,2,...,m) and their estimated covariance matrices.

Value

a list containing estimates of beta, sigma2, psi, an estimated covariance matrix for beta, the number of iterations actually performed, an indicator of whether the algorithm converged, and a vector of loglikelihood values at each iteration. If random.effects=T, also returns a matrix of estimated random effects (bhat) for individuals and an array of corresponding covariance matrices.

beta	vector of same length as "xcol" containing estimated fixed effects.
sigma2	estimate of error variance sigma2.
psi	matrix of dimension c(length(zcol),length(zcol)) containing the estimated covariance matrix psi.
converged	T if the algorithm converged, F if it did not
iter	number of iterations actually performed. Will be equal to "maxits" if converged=F.
loglik	vector of length "iter" reporting the value of the loglikelihood at each iteration.
cov.beta	matrix of dimension c(length(xcol),length(xcol)) containing estimated variances and covariances for elements of "beta".
bhat	if random.effects=T, a matrix with length(zcol) rows and m columns, where bhat[,i] is an empirical Bayes estimate of bi.
cov.b	if random.effects=T, an array of dimension length(zcol) by length(zcol) by m, where cov.b[,,i] is an empirical Bayes estimate of the covariance matrix associated with bi.

References

Schafer JL (1997) Imputation of missing covariates under a multivariate linear mixed model. Technical report 97-04, Dept. of Statistics, The Pennsylvania State University,

Schafer JL (2001). Multiple imputation with PAN. Chapter 12, pp357-77. of New Methods for the Analysis of Change. Edited by Collins LM, Sayer AG. American Psychological Association, Washington DC.

Schafer JL, Yucel RM (2002). Computational strategies for multivariate linear mixed-effects models with missing values. Journal of Computational and Graphical Statistics. 11:437-457

Examples

```
#########################################################################
# A simple linear model to these data using ecme(). This will be a
# traditional repeated-measures style additive model with a fixed effect
# for each column (occasion) and a random intercept for each subject.
#
# The data to be used is contained the object marijuana. Since the six
# measurements per subject were not clearly ordered in time, we consider
# a model that has an intercept and five dummy codes to allow the
# population means for the six occasions to be estimated freely together
# with an intercept randomly varied by subject. For a subject i with no
# missing values, the covariate matrices will be
#
#                   1 1 0 0 0 0              1
#                   1 0 1 0 0 0              1
#           Xi =    1 0 0 1 0 0       Zi =   1
#                   1 0 0 0 1 0              1
#                   1 0 0 0 0 1              1
#                   1 0 0 0 0 0              1
#
# When using ecme(), these are combined into a single matrix called
# pred. The pred matrix has length(y) rows. Each column of Xi and Zi
# must be represented in pred. Because Zi is merely the first column
# of Xi, we do not need to enter that column twice. So pred is simply
# the matrices Xi (i=1,...,9), stacked upon each other.
#
data(marijuana)
# we only use the complete data to illustrate
complete <- subset(marijuana,!is.na(y))
attach(complete)
pred <- with(complete,cbind(int,dummy1,dummy2,dummy3,dummy4,dummy5))
xcol <- 1:6
zcol <- 1
# Now we can fit the model.
result <- ecme(y,subj,occ,pred,xcol,zcol)
result

# Now we compare to lmer
if(require(lme4)) {
result <- lmer(y~-1+pred+(1|subj))
result
vcov(result)
detach(complete)
}
#########################################################################
```

---

marijuana *A pilot study of the clinical and psychological effects of maarijuana*

Description

Nine male subjects were given three treatments in the form of low-dose, high-dose, and placebo cigarettes. The order of treatments within subjects was balanced in a replicated 3 x 3 Latin square, but because the order for each subject was not reported in the article, (I shall proceed as if) the order effects are negligible. Changes in heart rate were recorded 15 and 90 minutes after marijuana use, and five of the 54 data values are missing.

Usage

```
data(marijuana)
```

Format

A data frame

Source

Wei AT, Zinberg NE, Nelson JM. Clinical and psychological effects of marijuana in man. Science 1968; 162:1234-1242

---

pan *Imputation of multivariate panel or cluster data*

Description

Gibbs sampler for the multivariate linear mixed model with incomplete data described by Schafer (1997). This function will typically be used to produce multiple imputations of missing data values in multivariate panel data or clustered data. The underlying model is

yi = Xi%*%beta + Zi%*%bi + ei, i=1,. . . ,m,

where

yi = (ni x r) matrix of incomplete multivariate data for subject or cluster i;

Xi = (ni x p) matrix of covariates;

Zi = (ni x q) matrix of covariates;

beta = (p x r) matrix of coefficients common to the population (fixed effects);

bi = (q x r) matrix of coefficients specific to subject or cluster i (random effects); and

ei = (ni x r) matrix of residual errors.

The matrix bi, when stacked into a single column, is assumed to be normally distributed with mean zero and unstructured covariance matrix psi, and the rows of ei are assumed to be independently normal with mean zero and unstructured covariance matrix sigma. Missing values may appear in yi in any pattern.

In most applications of this model, the first columns of Xi and Zi will be constant (one) and Zi will contain a subset of the columns of Xi.

Usage

```
pan(y, subj, pred, xcol, zcol, prior, seed, iter=1, start)
```

Arguments

y	matrix of responses. This is simply the individual yi matrices stacked upon one another. Each column of y corresponds to a response variable. Each row of y corresponds to a single subject-occasion, or to a single subject within a cluster. Missing values (NA) may occur in any pattern.
subj	vector of length nrow(y) giving the subject (or cluster) indicators i for the rows of y. For example, suppose that y is in fact rbind(y1,y2,y3,y4) where nrow(y1)=2, nrow(y2)=3, nrow(y3)=2, and nrow(y4)=7. Then subj should be c(1,1,2,2,2,3,3,4,4,4,4,4,4,4).
pred	matrix of covariates used to predict y. This should have the same number of rows as y. The first column will typically be constant (one), and the remaining columns correspond to other variables appearing in Xi and Zi.
xcol	vector of integers indicating which columns of pred will be used in Xi. That is, pred[,xcol] is the Xi matrices (stacked upon one another).
zcol	vector of integers indicating which columns of pred will be used in Zi. That is, pred[,zcol] is the Zi matrices (stacked upon one another).
prior	a list with four components (whose names are a, Binv, c, and Dinv, respectively) specifying the hyperparameters of the prior distributions for psi and sigma. For information on how to specify and interpret these hyperparameters, see Schafer (1997) and the example below. Note: This is a slight departure from the notation in Schafer (1997), where a and Binv were denoted by "nu1" and "Lambdainv1", and c and Dinv were "nu2" and "Lambdainv2".
seed	integer seed for initializing pan()'s internal random number generator. This argument should be a positive integer.
iter	total number of iterations or cycles of the Gibbs sampler to be carried out.
start	optional list of quantities to specify the initial state of the Gibbs sampler. This list has the same form as "last" (described below), one of the components returned by pan(). This argument allows the Gibbs sampler to be restarted from the final state of a previous run. If "start" is omitted then pan() chooses its own initial state.

Details

The Gibbs sampler algorithm used in pan() is described in detail by Schafer (1997).

Value

A list containing the following components. Note that when you are using pan() to produce multiple imputations, you will be primarily interested in the component "y" which contains the imputed data; the arrays "beta", "sigma", and "psi" will be used primarily for diagnostics (e.g. time-series plots) to assess the convergence behavior of the Gibbs sampler.

`beta` array of dimension c(length(xcol),ncol(y),iter) = (p x r x number of Gibbs cycles) containing the simulated values of beta from all cycles. That is, beta[,,T] is the (p x r) matrix of simulated fixed effects at cycle T.

`sigma` array of dimension c(ncol(y),ncol(y),iter) = (r x r x number of Gibbs cycles) containing the simulated values of sigma from all cycles. That is, sigma[,,T] is the simulated version of the model's sigma at cycle T.

`psi` array of dimension c(length(zcol)*ncol(y), length(zcol)*ncol(y), iter) = (q*r x q*r x number of Gibbs cycles) containing the simulated values of psi from all cycles. That is, psi[,,T] is the simulated version of the model's psi at cycle T.

`y` matrix of imputed data from the final cycle of the Gibbs sampler. Identical to the input argument y except that the missing values (NA) have been replaced by imputed values. If "iter" has been set large enough (which can be determined by examining time-series plots, etc. of "beta", "sigma", and "psi") then this is a proper draw from the posterior predictive distribution of the complete data.

`last` a list of four components characterizing the final state of the Gibbs sampler. The four components are: "beta", "sigma", "psi", and "y", which are the simulated values of the corresponding model quantities from the final cycle of Gibbs. This information is already contained in the other components returned by pan(); we are providing this list merely as a convenience, to allow the user to start future runs of the Gibbs sampler at this state.

Note

This function assumes that the rows of y (and thus the rows of subj and pred) have been sorted by subject number. That is, we assume that subj=sort(subj), y=y[order(subj),], and pred=pred[order(subj),]. If the matrix y is created by stacking yi, i=1,...,m then this will automatically be the case.

References

Schafer JL (1997) Imputation of missing covariates under a multivariate linear mixed model. Technical report 97-04, Dept. of Statistics, The Pennsylvania State University,

Schafer JL (2001). Multiple imputation with PAN. Chapter 12, pp357-77. of New Methods for the Analysis of Change. Edited by Collins LM, Sayer AG. American Psychological Association, Washington DC.

Schafer JL, Yucel RM (2002). Computational strategies for multivariate linear mixed-effects models with missing values. Journal of Computational and Graphical Statistics. 11:437-457

Examples

```
#############################################################################
# This example is somewhat atypical because the data consist of a
# single response variable (change in heart rate) measured repeatedly;
# most uses of pan() will involve r > 1 response variables. If we had
# r response variables rather than one, the only difference would be
# that the vector y below would become a matrix with r columns, one
# for each response variable. The dimensions of Sigma (the residual
# covariance matrix for the response) and Psi (the covariance matrix
# for the random effects) would also change to (r x r) and (r*q x r*q),
```

```
# respectively, where q is the number of random coefficients in the
# model (in this case q=1 because we have only random intercepts). The
# new dimensions for Sigma and Psi will be reflected in the prior
# distribution, as Dinv and Binv become (r x r) and (r*q x r*q).
#
# The pred matrix has the same number of rows as y, the number of
# subject-occasions. Each column of Xi and Zi must be represented in
# pred. Because Zi is merely the first column of Xi, we do not need to
# enter that column twice. So pred is simply the matrix Xi, stacked
# upon itself nine times.
#
data(marijuana)
attach(marijuana)
pred <- with(marijuana,cbind(int,dummy1,dummy2,dummy3,dummy4,dummy5))
#
# Now we must tell pan that all six columns of pred are to be used in
# Xi, but only the first column of pred appears in Zi.
#
xcol <- 1:6
zcol <- 1
############################################################################
# The model specification is now complete. The only task that remains
# is to specify the prior distributions for the covariance matrices
# Sigma and Psi.
#
# Recall that the dimension of Sigma is (r x r) where r
# is the number of response variables (in this case, r=1). The prior
# distribution for Sigma is inverted Wishart with hyperparameters a
# (scalar) and Binv (r x r), where a is the imaginary degrees of freedom
# and Binv/a is the prior guesstimate of Sigma. The value of a must be
# greater than or equal to r. The "least informative" prior possible
# would have a=r, so here we will take a=1. As a prior guesstimate of
# Sigma we will use the (r x r) identity matrix, so Binv = 1*1 = 1.
#
# By similar reasoning we choose the prior distribution for Psi. The
# dimension of Psi is (r*q x r*q) where q is the number of random
# effects in the model (i.e. the length of zcol, which in this case is
# one). The hyperparameters for Psi are c and Dinv, where c is the
# imaginary degrees of freedom (which must be greater than or equal to
# r*q) and Dinv/c is the prior guesstimate of Psi. We will take c=1
# and Dinv=1*1 = 1.
#
# The prior is specified as a list with four components named a, Binv,
# c, and Dinv, respectively.
#
prior <- list(a=1,Binv=1,c=1,Dinv=1)
############################################################################
# Now we are ready to run pan(). Let's assume that the pan function
# and the object code have already been loaded into R. First we
# do a preliminary run of 1000 iterations.
#
result <- pan(y,subj,pred,xcol,zcol,prior,seed=13579,iter=1000)
#
```

```
# Check the convergence behavior by making time-series plots and acfs
# for the model parameters. Variances will be plotted on a log
# scale. We'll assume that a graphics device has already been opened.
#
plot(1:1000,log(result$sigma[1,1,]),type="l")
acf(log(result$sigma[1,1,]))
plot(1:1000,log(result$psi[1,1,]),type="l")
acf(log(result$psi[1,1,]))
par(mfrow=c(3,2))
for(i in 1:6) plot(1:1000,result$beta[i,1,],type="l")
for(i in 1:6) acf(result$beta[i,1,])
#
# This example appears to converge very rapidly; the only appreciable
# autocorrelations are found in Psi, and even those die down by lag
# 10. With a sample this small we can afford to be cautious, so let's
# impute the missing data m=10 times taking 100 steps between
# imputations. We'll use the current simulated value of y as the first
# imputation, then restart the chain where we left off to produce
# the second through the tenth.
#
y1 <- result$y
result <- pan(y,subj,pred,xcol,zcol,prior,seed=9565,iter=100,start=result$last)
y2 <- result$y
result <- pan(y,subj,pred,xcol,zcol,prior,seed=6047,iter=100,start=result$last)
y3 <- result$y
result <- pan(y,subj,pred,xcol,zcol,prior,seed=3955,iter=100,start=result$last)
y4 <- result$y
result <- pan(y,subj,pred,xcol,zcol,prior,seed=4761,iter=100,start=result$last)
y5 <- result$y
result <- pan(y,subj,pred,xcol,zcol,prior,seed=9188,iter=100,start=result$last)
y6 <- result$y
result <- pan(y,subj,pred,xcol,zcol,prior,seed=9029,iter=100,start=result$last)
y7 <- result$y
result <- pan(y,subj,pred,xcol,zcol,prior,seed=4343,iter=100,start=result$last)
y8 <- result$y
result <- pan(y,subj,pred,xcol,zcol,prior,seed=2372,iter=100,start=result$last)
y9 <- result$y
result <- pan(y,subj,pred,xcol,zcol,prior,seed=7081,iter=100,start=result$last)
y10 <- result$y
##############################################################################
# Now we combine the imputation results according to mitools
##############################################################################
# First, we build data frames from above,
d1 <- data.frame(y=y1,subj,pred)
d2 <- data.frame(y=y2,subj,pred)
d3 <- data.frame(y=y3,subj,pred)
d4 <- data.frame(y=y4,subj,pred)
d5 <- data.frame(y=y5,subj,pred)
d6 <- data.frame(y=y6,subj,pred)
d7 <- data.frame(y=y7,subj,pred)
d8 <- data.frame(y=y8,subj,pred)
d9 <- data.frame(y=y9,subj,pred)
d10 <- data.frame(y=y10,subj,pred)
```

```
# Second, we establish a S3 object as needed for the function MIcombine
# nevertheless we start with an ordinary least squares regression
require(mitools)
d <- imputationList(list(d1,d2,d3,d4,d5,d6,d7,d8,d9,d10))
w <- with(d,lm(y~-1+pred))
MIcombine(w)
# Now, we can turn to lmer as in lme4 package but in this case it is the
# same.
if(require(lme4)) {
w2 <- with(d,lmer(y~-1+pred+(1|subj)))
b <- MIextract(w2,fun=fixef)
Var <- function(obj) unlist(lapply(diag(vcov(obj)),function(m) m))
v <- MIextract(w2,fun=Var)
MIcombine(b,v)
detach(marijuana)
}
### bivariate example

data(bitest)
attach(bitest)
y <- with(bitest,cbind(y1,y2))

subj <- c(clusterid)
pred <- cbind (int, x1, x2, x3)
xcol <- 1:4
zcol <- 1
a <- 2
c <- 2
id2 <- matrix(c(1,0,0,1),ncol=2,nrow=2)
Binv <- a*id2
Dinv <- c*id2
prior <- list(a=a, Binv=Binv, c=c, Dinv=Dinv)
result <- pan(y, subj, pred, xcol, zcol, prior, seed=12345, iter=1000)
```

---

pan.bd *Imputation of multivariate panel or cluster data*

Description

Implementation of pan() that restricts the covariance matrix for the random effects to be block-diagonal. This function is identical to pan() in every way except that psi is now characterized by a set of r matrices of dimension q x q.

Usage

```
pan.bd(y, subj, pred, xcol, zcol, prior, seed, iter=1, start)
```

Arguments

y	See description for pan().
subj	See description for pan().
pred	See description for pan().
xcol	See description for pan().
zcol	See description for pan().
prior	Same as for pan() except that the hyperparameters for psi have new dimensions. The hyperparameter c is now a vector of length r, where c[j] contains the prior degrees of freedom for the jth block portion of psi (j=1,. . . ,r). The hyperparameter Dinv is now an array of dimension c(q,q,r), where Dinv[,,j] contains the prior scale matrix for the jth block portion of psi (j=1,. . . ,r).
seed	See description for pan().
iter	See description for pan().
start	See description for pan().

Value

A list with the same components as that from pan(), with two minor differences: the dimension of "psi" is now (q x q x r x "iter"), and the dimension of "last\$psi" is now (q x q x r).

YWC.data *A data frame for multivariate imputation*

Description

The data serves as an worked example for PAN as follows,.

```
head(YWC.data)
Yi.Matrix <- as.matrix(YWC.data[,4:39])
Subj.vector <- as.vector(as.matrix(YWC.data[,2]))
YWC.data$intercept <- 1.
Pred.Matrix <- as.matrix(YWC.data[,c(40,3)])
xcol <- 1:2
zcol <- 1
a.parameter = ncol(Yi.Matrix)
Binv.parameter = diag(a.parameter*a.parameter)
c.parameter = (zcol*ncol(Yi.Matrix))
Dinv.parameter = diag(c.parameter*c.parameter)
Prior.Distribution <- list(a = a.parameter,
                           Binv = Binv.parameter,
                           c = c.parameter,
                           Dinv = Dinv.parameter)
Imputation.Result <- pan(Yi.Matrix, Subj.vector, Pred.Matrix, xcol, zcol, Prior.Distribution, seed=44
```

Usage

```
data(YWC.data)
```

Format

A data frame

Source

Steven J. Pierce

Index

∗Topic **datasets**
bitest, 2
marijuana, 6
YWC.data, 13
∗Topic **models**
ecme, 3
pan, 7
pan.bd, 12

bitest, 2

ecme, 3

marijuana, 6

pan, 7
pan.bd, 12

YWC.data, 13