Package ‘QRegVCM’

March 16, 2018

Type Package

Title Quantile Regression in Varying-Coefficient Models

Version 1.2

Date 2018-02-26

Author 'Andriyana, Y.' [aut, cre], 'Ibrahim M. A.' [aut], 'Gijbels, I.' [ctb], 'Verhasselt A.' [ctb]

Maintainer ``Andriyana.Y'' <y.andriyana@unpad.ac.id>

Description Quantile regression in varying-coefficient models (VCM) using one particular nonparametric technique called P-splines. The functions can be applied on three types of VCM; (1) Homoscedastic VCM, (2) Simple heteroscedastic VCM, and (3) General heteroscedastic VCM.

Depends R (>= 3.4.0), quantreg, SparseM, truncSP

License GPL-2

NeedsCompilation no

Repository CRAN

Date/Publication 2018-03-16 10:02:26 UTC

R topics documented:

AHeVT — *AHe V(t)-approach*

Description

The adapted He (1997) approach considering a simple hetersocedastic varying-coefficient model, V(t).

$$Y(t) = \sum_{k=0}^{p} \beta_k(t) X^{(k)}(t) + V(t)\varepsilon(t)$$

.

Usage

```
AHeVT(VecX, times, subj, X, y, d, tau, kn, degree, lambda, gam)
```

Arguments

`VecX`	The representative values for each covariate used to estimate the desired conditional quantile curves.
`times`	The vector of time variable.
`subj`	The vector of subjects/individuals.
`X`	The covariate containing 1 as its first component (including intercept in the model)
`y`	The response vector.
`d`	The order of differencing operator for each covariate.
`tau`	The quantiles of interest.
`kn`	The number of knots for each covariate.
`degree`	The degree of B-spline basis for each covariate.
`lambda`	The grid of smoothing parameter to control the trade of between fidelity and penalty term (use a fine grid of lambda).
`gam`	The power used in estimating the smooting parameter for each covariate (e.g. gam=1 or gam=0.5).

Value

`hat_bt50`	The median coefficients estimators.
`hat_VT`	The variability estimator.
`C`	The estimators of the tau-th quantile of the estimated residuals.
`qhat`	The conditional quantile curves estimator.

Note

Some warning messages are related to the function `rq.fit.sfn`.

Author(s)

Yudhie Andriyana

References

Andriyana, Y., Gijbels, I., and Verhasselt, A. P-splines quantile regression estimation in varying coefficient models. *Test* 23, 1 (2014a),153–194.

Andriyana, Y., Gijbels, I. and Verhasselt, A. (2014b). Quantile regression in varying coefficient models: non-crossingness and heteroscedasticity. *Manuscript*.

He, X. (1997). Quantile curves without crossing. *The American Statistician*, 51, 186–192.

See Also

rq.fit.sfn as.matrix.csr truncSP

Examples

```
data(PM10)

PM10 = PM10[order(PM10$day,PM10$hour,decreasing=FALSE),]

y = PM10$PM10[1:200]
times = PM10$hour[1:200]
subj = PM10$day[1:200]
dim = length(y)
x0 = rep(1,200)
x1 = PM10$cars[1:200]
x2 = PM10$wind.speed[1:200]

X = cbind(x0, x1, x2)

VecX = c(1, max(x1), max(x2))

##########################
#### Input parameters ####
##########################
kn = c(10, 10, 10)
degree = c(3, 3, 3)
taus = seq(0.1,0.9,0.1)
lambdas = c(1,1.5,2)
d = c(1, 1, 1)
gam = 1/2
##########################


AHe = AHeVT(VecX=VecX, times=times, subj=subj, X=X, y=y, d=d, tau=taus,
      kn=kn, degree=degree, lambda=lambdas, gam=gam)
hat_bt50 = AHe$hat_bt50
hat_VT = AHe$hat_Vt
C = AHe$C
```

```
qhat = AHe$qhat

qhat1 = qhat[,1]
qhat2 = qhat[,2]
qhat3 = qhat[,3]
qhat4 = qhat[,4]
qhat5 = qhat[,5]
qhat6 = qhat[,6]
qhat7 = qhat[,7]
qhat8 = qhat[,8]
qhat9 = qhat[,9]

hat_bt0 = hat_bt50[seq(1,dim)]
hat_bt1 = hat_bt50[seq((dim+1),(2*dim))]
hat_bt2 = hat_bt50[seq((2*dim+1),(3*dim))]

i = order(times, hat_VT, qhat1, qhat2, qhat3, qhat4, qhat5, qhat6, qhat7,
    qhat8, qhat9, hat_bt0, hat_bt1, hat_bt2);
times = times[i]; hat_VT=hat_VT[i]; qhat1 = qhat1[i]; qhat2=qhat2[i];
qhat3=qhat3[i]; qhat4=qhat4[i]; qhat5=qhat5[i]; qhat6=qhat6[i];
qhat7=qhat7[i]; qhat8=qhat8[i]; qhat9=qhat9[i];
hat_bt0=hat_bt0[i];  hat_bt1=hat_bt1[i];  hat_bt2=hat_bt2[i];


### Plot coefficients

plot(hat_bt0~times, lwd=2, type="l", xlab="hour", ylab="baseline PM10");
plot(hat_bt1~times, lwd=2, type="l", xlab="hour",
    ylab="coefficient of cars");
plot(hat_bt2~times, lwd=2, type="l", xlab="hour",
    ylab="coefficient of wind");


### Plot variability V(t)

plot(hat_VT~times, ylim=c(min(hat_VT), max(hat_VT)), xlab="hour",
    ylab="", type="l", lwd=2);
mtext(expression(hat(V)(t)), side=2, cex=1, line=3)

### Plot conditional quantiles estimators

ylim = c(-4, 6)
plot(qhat1~times, col="magenta", cex=0.2, lty=5, lwd=2, type="l",
    ylim=ylim, xlab="hour", ylab="PM10");
lines(qhat2~times, col="aquamarine4", cex=0.2, lty=4, lwd=2);
lines(qhat3~times, col="blue", cex=0.2, lty=3, lwd=3);
lines(qhat4~times, col="brown", cex=0.2, lty=2, lwd=2);
lines(qhat5~times, col="black", cex=0.2, lty=1, lwd=2);
lines(qhat6~times, col="orange", cex=0.2, lty=2, lwd=2)
lines(qhat7~times, col="darkcyan", cex=0.2, lty=3, lwd=3);
lines(qhat8~times, col="green", cex=0.2, lty=4, lwd=2);
lines(qhat9~times, col="red", cex=0.2, lty=5, lwd=3)
```

```
legend("bottom", c(expression(tau==0.9), expression(tau==0.8),
    expression(tau==0.7), expression(tau==0.6), expression(tau==0.5),
    expression(tau==0.4), expression(tau==0.3), expression(tau==0.2),
    expression(tau==0.1)), ncol=1, col=c("red","green","darkcyan",
    "orange","black","brown","blue","aquamarine4","magenta"),
    lwd=c(2,2,3,2,2,2,3,2,2), lty=c(5,4,3,2,1,2,3,4,5))
```

`AHeVXT` *AHe V(X(t),t)-approach*

Description

The adapted He (1997) approach considering a general hetersocedastic varying-coefficient model, V(X(t),t).

$$Y(t) = \sum_{k=0}^{p} \beta_k(t) X^{(k)}(t) + V(X(t),t)\varepsilon(t)$$

where

$$V(X(t),t) = \sum_{k=0}^{p} \gamma_k(t) X^{(k)}(t)$$

.

Usage

```
AHeVXT(VecX, times, subj, X, y, d, tau, kn, degree, lambda, gam)
```

Arguments

`VecX`	The representative values for each covariate used to estimate the desired conditional quantile curves.
`times`	The vector of the time variable.
`subj`	The vector of subjects/individuals.
`X`	The covariate matrix containing 1 as its first column (including intercept in the model).
`y`	The response vector.
`d`	The order of the differencing operator for each covariate.
`tau`	The quantiles of interest.
`kn`	The number of knots for each covariate.
`degree`	The degree of the B-spline basis function for each covariate.
`lambda`	The grid for the smoothing parameter to control the trade of between fidelity and penalty term (use a fine grid of lambda).
`gam`	The power used in estimating the smooting parameter for each covariate (e.g. gam=1 or gam=0.5).

Value

hat_bt50	The median coefficients estimators.
hat_gt50	The median coefficients estimators for the variability function V(X(t),t).
hat_VXT	The variability estimator.
C	The estimators of the tau-th quantile of the estimated residuals.
qhat	The conditional quantile curves estimator.

Note

Some warning messages are related to the function `rq.fit.sfn`.

Author(s)

Yudhie Andriyana

References

Andriyana, Y., Gijbels, I., and Verhasselt, A. P-splines quantile regression estimation in varying coefficient models. *Test* 23, 1 (2014a),153–194.

Andriyana, Y., Gijbels, I. and Verhasselt, A. (2014b). Quantile regression in varying coefficient models: non-crossingness and heteroscedasticity. *Manuscript*.

He, X. (1997). Quantile curves without crossing. *The American Statistician*, 51, 186–192.

See Also

`rq.fit.sfn` `as.matrix.csr` `truncSP`

Examples

```
data(PM10)

PM10 = PM10[order(PM10$day,PM10$hour,decreasing=FALSE),]

y = PM10$PM10[1:200]
times = PM10$hour[1:200]
subj = PM10$day[1:200]
dim = length(y)
x0 = rep(1,200)
x1 = PM10$cars[1:200]
x2 = PM10$wind.speed[1:200]

X = cbind(x0, x1, x2)

VecX = c(1, max(x1), max(x2))

##########################
#### Input parameters ####
##########################
```

```
kn = c(10, 10, 10)
degree = c(3, 3, 3)
taus = seq(0.1,0.9,0.1)
lambdas = c(1,1.5,2)
d = c(1, 1, 1)
gam = 1/2
##########################

AHe = AHeVXT(VecX=VecX, times=times, subj=subj, X=X, y=y, d=d,
      tau=taus,  kn=kn, degree=degree, lambda=lambdas, gam=gam)
hat_bt50 = AHe$hat_bt50
hat_gt50 = AHe$hat_gt50
hat_VXT = AHe$hat_VXT
C = AHe$C
qhat = AHe$qhat

qhat1 = qhat[,1]
qhat2 = qhat[,2]
qhat3 = qhat[,3]
qhat4 = qhat[,4]
qhat5 = qhat[,5]
qhat6 = qhat[,6]
qhat7 = qhat[,7]
qhat8 = qhat[,8]
qhat9 = qhat[,9]

hat_bt0 = hat_bt50[seq(1,dim)]
hat_bt1 = hat_bt50[seq((dim+1),(2*dim))]
hat_bt2 = hat_bt50[seq((2*dim+1),(3*dim))]

hat_gt0 = hat_gt50[seq(1,dim)]
hat_gt1 = hat_gt50[seq((dim+1),(2*dim))]
hat_gt2 = hat_gt50[seq((2*dim+1),(3*dim))]

i = order(times, hat_VXT, qhat1, qhat2, qhat3, qhat4, qhat5, qhat6, qhat7,
    qhat8, qhat9,
hat_bt0, hat_bt1, hat_bt2, hat_gt0, hat_gt1, hat_gt2);
times = times[i]; hat_VXT=hat_VXT[i]; qhat1 = qhat1[i]; qhat2=qhat2[i];
qhat3=qhat3[i]; qhat4=qhat4[i]; qhat5=qhat5[i]; qhat6=qhat6[i];
qhat7=qhat7[i]; qhat8=qhat8[i]; qhat9=qhat9[i];
hat_bt0=hat_bt0[i];  hat_bt1=hat_bt1[i];  hat_bt2=hat_bt2[i];
hat_gt0=hat_gt0[i];  hat_gt1=hat_gt1[i];  hat_gt2=hat_gt2[i];


### Plot coefficients

plot(hat_bt0~times, lwd=2, type="l", xlab="hour", ylab="baseline PM10");
plot(hat_bt1~times, lwd=2, type="l", xlab="hour",
    ylab="coefficient of cars");
plot(hat_bt2~times, lwd=2, type="l", xlab="hour",
    ylab="coefficient of wind");

###
```

```
plot(hat_gt0~times, lwd=2, type="l", xlab="hour", ylab="baseline PM10");
plot(hat_gt1~times, lwd=2, type="l", xlab="hour",
    ylab="coefficient of cars");
plot(hat_gt2~times, lwd=2, type="l", xlab="hour",
    ylab="coefficient of wind");

### Plot variability V(X(t),t)

plot(hat_VXT~times, ylim=c(min(hat_VXT), max(hat_VXT)), xlab="hour", ylab="");
mtext(expression(hat(V)(X(t),t)), side=2, cex=1, line=3)

### Plot conditional quantiles estimators

ylim = range(qhat1, qhat9)
ylim = c(-4, 6)
plot(qhat1~times, col="magenta", cex=0.2, lty=5, lwd=2, type="l", ylim=ylim,
    xlab="hour", ylab="PM10");
lines(qhat2~times, col="aquamarine4", cex=0.2, lty=4, lwd=2);
lines(qhat3~times, col="blue", cex=0.2, lty=3, lwd=3);
lines(qhat4~times, col="brown", cex=0.2, lty=2, lwd=2);
lines(qhat5~times, col="black", cex=0.2, lty=1, lwd=2);
lines(qhat6~times, col="orange", cex=0.2, lty=2, lwd=2)
lines(qhat7~times, col="darkcyan", cex=0.2, lty=3, lwd=3);
lines(qhat8~times, col="green", cex=0.2, lty=4, lwd=2);
lines(qhat9~times, col="red", cex=0.2, lty=5, lwd=3)

legend("bottom", c(expression(tau==0.9), expression(tau==0.8),
      expression(tau==0.7), expression(tau==0.6), expression(tau==0.5),
      expression(tau==0.4), expression(tau==0.3), expression(tau==0.2),
      expression(tau==0.1)), ncol=3, col=c("red","green","darkcyan","orange",
      "black","brown","blue","aquamarine4","magenta"), lwd=c(2,2,3,2,2,2,3,2,2),
      lty=c(5,4,3,2,1,2,3,4,5))
```

CD4 *The CD4 dataset*

Description

This dataset is a subset of the Multicenter AIDS cohort study. The data contain repeated measurements of physical examinations and CD4 percentages of 283 homosexual men who became HIV-positive between year 1984 and 1991.

Usage

```
CD4
```

Format

data frame with 6 variables and 1817 observations.

Details

It contains the following variables:

subj the subject (man) indicator

Time the time in years

Smooking the smoking status

Age age at HIV infection

PreCD4 the pre-infection CD4 percentage

CD4 the CD4 percentage

`QRIndiv` *Individual quantile objective function*

Description

The estimation of conditional quantile curves using individual quantile objective function.

Usage

```
QRIndiv(VecX, tau, times, subj, X, y, d, kn, degree, lambda, gam)
```

Arguments

`VecX`	The representative values for each covariate used to estimate the desired conditional quantile curves.
`tau`	The quantiles of interest.
`times`	The vector of the time variable.
`subj`	The vector of subjects/individuals.
`X`	The covariate matrix containing 1 as its first column (including intercept in the model).
`y`	The response vector.
`d`	The order of the differencing operator for each covariate.
`kn`	The number of knots for each covariate.
`degree`	The degree of the B-spline basis function for each covariate.
`lambda`	The grid for the smoothing parameter to control the trade of between fidelity and penalty term (use a fine grid of lambda).
`gam`	The power used in estimating the smooting parameter for each covariate (e.g. gam=1 or gam=0.5).

Value

alpha	The estimator of the coefficient vector of the basis B-splines.
hat_bt	The varying coefficients estimators.
qhat	The conditional quantile curves estimator.

Note

Some warning messages are related to the function `rq.fit.sfn`.

Author(s)

Yudhie Andriyana

References

Andriyana, Y., Gijbels, I., and Verhasselt, A. P-splines quantile regression estimation in varying coefficient models. *Test* 23, 1 (2014a),153–194.

See Also

`rq.fit.sfn` `as.matrix.csr` `truncSP`

Examples

```
data(PM10)

PM10 = PM10[order(PM10$day,PM10$hour,decreasing=FALSE),]

y = PM10$PM10[1:200]
times = PM10$hour[1:200]
subj = PM10$day[1:200]
dim = length(y)
x0 = rep(1,200)
x1 = PM10$cars[1:200]
x2 = PM10$wind.speed[1:200]

X = cbind(x0, x1, x2)

VecX = c(1, max(x1), max(x2))

##########################
#### Input parameters ####
##########################
kn = c(10, 10, 10)
degree = c(3, 3, 3)
taus = seq(0.1,0.9,0.1)
lambdas = c(1,1.5,2)
d = c(1, 1, 1)
gam = 1/2
##########################
```

```
qhat = QRIndiv(VecX=VecX, tau=taus, times=times, subj=subj, X=X,
y=y, d=d, kn=kn, degree=degree, lambda=lambdas, gam=gam)$qhat

qhat1 = qhat[,1]
qhat2 = qhat[,2]
qhat3 = qhat[,3]
qhat4 = qhat[,4]
qhat5 = qhat[,5]
qhat6 = qhat[,6]
qhat7 = qhat[,7]
qhat8 = qhat[,8]
qhat9 = qhat[,9]


i = order(times, y, qhat1, qhat2, qhat3, qhat4, qhat5, qhat6, qhat7,
    qhat8, qhat9);

times = times[i]; y = y[i]; qhat1 = qhat1[i]; qhat2=qhat2[i];
qhat3=qhat3[i]; qhat4=qhat4[i]; qhat5=qhat5[i]; qhat6=qhat6[i];
qhat7=qhat7[i]; qhat8=qhat8[i]; qhat9=qhat9[i];

ylim = range(qhat1, qhat9)
plot(qhat1~times, col="magenta", cex=0.2, lty=5, lwd=2, type="l",
    ylim=ylim, xlab="hour", ylab="PM10");
lines(qhat2~times, col="aquamarine4", cex=0.2, lty=4, lwd=2);
lines(qhat3~times, col="blue", cex=0.2, lty=3, lwd=3);
lines(qhat4~times, col="brown", cex=0.2, lty=2, lwd=2);
lines(qhat5~times, col="black", cex=0.2, lty=1, lwd=2);
lines(qhat6~times, col="orange", cex=0.2, lty=2, lwd=2)
lines(qhat7~times, col="darkcyan", cex=0.2, lty=3, lwd=3);
lines(qhat8~times, col="green", cex=0.2, lty=4, lwd=2);
lines(qhat9~times, col="red", cex=0.2, lty=5, lwd=3)

legend("bottom", c(expression(tau==0.9), expression(tau==0.8),
      expression(tau==0.7), expression(tau==0.6), expression(tau==0.5),
      expression(tau==0.4), expression(tau==0.3), expression(tau==0.2),
      expression(tau==0.1)), ncol=1, col=c("red","green","darkcyan",
      "orange","black","brown","blue","aquamarine4","magenta"),
      lwd=c(2,2,3,2,2,2,3,2,2), lty=c(5,4,3,2,1,2,3,4,5))
```

`QRSimul` *Unweighted simultaneous objective function*

Description

The etimation of conditional quantile curves using unweighted simultaneous objective function involving non-crossing constraints.

Usage

```
QRSimul(VecX, tau, times, subj, X, y, d, kn, degree, lambda, gam)
```

Arguments

VecX	The representative values for each covariate used to estimate the desired conditional quantile curves.
tau	The quantiles of interest.
times	The vector of the time variable.
subj	The vector of subjects/individuals.
X	The covariate matrix containing 1 as its first column (including intercept in the model).
y	The response vector.
d	The order of the differencing operator for each covariate.
kn	The number of knots for each covariate.
degree	The degree of the B-spline basis function for each covariate.
lambda	The grid for the smoothing parameter to control the trade of between fidelity and penalty term (use a fine grid of lambda).
gam	The power used in estimating the smooting parameter for each covariate (e.g. gam=1 or gam=0.5).

Value

W	The weight for each subject corresponding to the length of its repeated measurement.
alpha	The estimators of the coefficient vector of the basis B-splines.
hat_bt0	The baseline estimators.
hat_btk	The varying coefficient estimators.
qhat_h	The estimators of the τ_h-th conditional quantile curves.

Note

Some warning messages are related to the function `rq.fit.sfn`.

Author(s)

Yudhie Andriyana

References

Andriyana, Y., Gijbels, I., and Verhasselt, A. P-splines quantile regression estimation in varying coefficient models. *Test* 23, 1 (2014a),153–194.

Andriyana, Y., Gijbels, I. and Verhasselt, A. (2014b). Quantile regression in varying coefficient models: non-crossingness and heteroscedasticity. *Manuscript*.

See Also

rq.fit.sfn as.matrix.csr truncSP

Examples

```
data(PM10)

PM10 = PM10[order(PM10$day,PM10$hour,decreasing=FALSE),]

y = PM10$PM10[1:200]
times = PM10$hour[1:200]
subj = PM10$day[1:200]
dim = length(y)
x0 = rep(1,200)
x1 = PM10$cars[1:200]
x2 = PM10$wind.speed[1:200]

X = cbind(x0, x1, x2)

VecX = c(1, max(x1), max(x2))

##########################
#### Input parameters ####
##########################
kn = c(10, 10, 10)
degree = c(3, 3, 3)
taus = seq(0.1,0.9,0.1)
lambdas = c(1,1.5,2)
d = c(1, 1, 1)
gam = 1
##########################


Simul = QRSimul(VecX=VecX, tau=taus, times=times, subj=subj, X=X, y=y,
        d=d, kn=kn, degree=degree, lambda=lambdas, gam=gam)

hat_bt0 = Simul$hat_bt0
hat_btk = Simul$hat_btk
qhat = Simul$qhat

qhat1 = qhat[,1]
qhat2 = qhat[,2]
qhat3 = qhat[,3]
qhat4 = qhat[,4]
qhat5 = qhat[,5]
qhat6 = qhat[,6]
qhat7 = qhat[,7]
qhat8 = qhat[,8]
qhat9 = qhat[,9]
```

```
i = order(times, y, qhat1, qhat2, qhat3, qhat4, qhat5, qhat6, qhat7,
    qhat8, qhat9);

times = times[i]; y = y[i]; qhat1 = qhat1[i]; qhat2=qhat2[i];
qhat3=qhat3[i]; qhat4=qhat4[i]; qhat5=qhat5[i]; qhat6=qhat6[i];
qhat7=qhat7[i]; qhat8=qhat8[i]; qhat9=qhat9[i];

ylim = range(qhat1, qhat9)
ylim = c(-4, 6)
plot(qhat1~times, col="magenta", cex=0.2, lty=5, lwd=2, type="l", ylim=ylim,
xlab="hour", ylab="PM10");
lines(qhat2~times, col="aquamarine4", cex=0.2, lty=4, lwd=2);
lines(qhat3~times, col="blue", cex=0.2, lty=3, lwd=3);
lines(qhat4~times, col="brown", cex=0.2, lty=2, lwd=2);
lines(qhat5~times, col="black", cex=0.2, lty=1, lwd=2);
lines(qhat6~times, col="orange", cex=0.2, lty=2, lwd=2)
lines(qhat7~times, col="darkcyan", cex=0.2, lty=3, lwd=3);
lines(qhat8~times, col="green", cex=0.2, lty=4, lwd=2);
lines(qhat9~times, col="red", cex=0.2, lty=5, lwd=3)

legend("bottom", c(expression(tau==0.9), expression(tau==0.8),
    expression(tau==0.7), expression(tau==0.6), expression(tau==0.5),
    expression(tau==0.4), expression(tau==0.3), expression(tau==0.2),
    expression(tau==0.1)), ncol=1, col=c("red","green","darkcyan",
    "orange","black","brown","blue","aquamarine4","magenta"),
    lwd=c(2,2,3,2,2,2,3,2,2), lty=c(5,4,3,2,1,2,3,4,5))
```

`QRStepwise` *Stepwise procedure*

Description

The etimation of conditional quantile curves step-by-step involving the non-crossing constraints.

Usage

```
QRStepwise(VecX, tau, times, subj, X, y, d, kn, degree, lambda, gam)
```

Arguments

`VecX`	The representative values for each covariate used to estimate the desired conditional quantile curves.
`tau`	The quantiles of interest.

`times`	The vector of the time variable.
`subj`	The vector of subjects/individuals.
`X`	The covariate matrix containing 1 as its first column (including intercept in the model).
`y`	The response vector.
`d`	The order of the differencing operator for each covariate.
`kn`	The number of knots for each covariate.
`degree`	The degree of the B-spline basis function for each covariate.
`lambda`	The grid for the smoothing parameter to control the trade of between fidelity and penalty term (use a fine grid of lambda).
`gam`	The power used in estimating the smooting parameter for each covariate (e.g. gam=1 or gam=0.5).

Value

`alpha`	The estimators of the coefficient vector of the basis B-splines.
`hat_bt`	The varying-coefficient estimators.
`W`	The weight for each subject corresponding to the length of its repeated measurement
`qhat`	The conditional quantile curves estimator.

Note

Some warning messages are related to the function `rq.fit.sfn`.

Author(s)

Yudhie Andriyana

References

Andriyana, Y., Gijbels, I., and Verhasselt, A. P-splines quantile regression estimation in varying coefficient models. *Test* 23, 1 (2014a),153–194.

Andriyana, Y., Gijbels, I. and Verhasselt, A. (2014b). Quantile regression in varying coefficient models: non-crossingness and heteroscedasticity. *Manuscript*.

Wu, Y. and Liu, Y. Stepwise multiple quantile regression estimation using non-crossing constraints. *Statistics and Its Interface* 2, (2009), 299–310.

See Also

`rq.fit.sfn` `as.matrix.csr` `truncSP`

Examples

```
data(PM10)

PM10 = PM10[order(PM10$day,PM10$hour,decreasing=FALSE),]

y = PM10$PM10[1:200]
time_ub = PM10$hour[1:200]
subj = PM10$day[1:200]
dim = length(y)
x0 = rep(1,200)
x1 = PM10$cars[1:200]
x2 = PM10$wind.speed[1:200]

X = cbind(x0, x1, x2)

VecX = c(1, max(x1), max(x2))

##########################
#### Input parameters ####
##########################
kn = c(10, 10, 10)
degree = c(3, 3, 3)
taus = seq(0.1,0.9,0.1)
lambdas = c(1,1.5,2)
d = c(1, 1, 1)
gam = 1
##########################


Step = QRStepwise(VecX=VecX, tau=taus, time_ub, subj, X, y, d, kn, degree,
       lambda=lambdas, gam=gam)

qhat = Step$qhat

qhat1 = qhat[,1]
qhat2 = qhat[,2]
qhat3 = qhat[,3]
qhat4 = qhat[,4]
qhat5 = qhat[,5]
qhat6 = qhat[,6]
qhat7 = qhat[,7]
qhat8 = qhat[,8]
qhat9 = qhat[,9]


i = order(time_ub, y, qhat1, qhat2, qhat3, qhat4, qhat5, qhat6, qhat7,
    qhat8, qhat9);

time_ub = time_ub[i]; y = y[i]; qhat1 = qhat1[i]; qhat2=qhat2[i];
qhat3=qhat3[i]; qhat4=qhat4[i]; qhat5=qhat5[i]; qhat6=qhat6[i];
qhat7=qhat7[i]; qhat8=qhat8[i]; qhat9=qhat9[i];
```

```
ylim = range(qhat1, qhat9)
ylim = c(-4, 6)
plot(qhat1~time_ub, col="magenta", cex=0.2, lty=5, lwd=2, type="l", ylim=ylim,
    xlab="hour", ylab="PM10");
lines(qhat2~time_ub, col="aquamarine4", cex=0.2, lty=4, lwd=2);
lines(qhat3~time_ub, col="blue", cex=0.2, lty=3, lwd=3);
lines(qhat4~time_ub, col="brown", cex=0.2, lty=2, lwd=2);
lines(qhat5~time_ub, col="black", cex=0.2, lty=1, lwd=2);
lines(qhat6~time_ub, col="orange", cex=0.2, lty=2, lwd=2)
lines(qhat7~time_ub, col="darkcyan", cex=0.2, lty=3, lwd=3);
lines(qhat8~time_ub, col="green", cex=0.2, lty=4, lwd=2);
lines(qhat9~time_ub, col="red", cex=0.2, lty=5, lwd=3)

legend("bottom", c(expression(tau==0.9), expression(tau==0.8),
    expression(tau==0.7), expression(tau==0.6), expression(tau==0.5),
    expression(tau==0.4), expression(tau==0.3), expression(tau==0.2),
    expression(tau==0.1)), ncol=1, col=c("red","green","darkcyan",
    "orange","black","brown","blue","aquamarine4","magenta"),
    lwd=c(2,2,3,2,2,2,3,2,2), lty=c(5,4,3,2,1,2,3,4,5))
```

QRWSimul *Weighted simultaneous objective function*

Description

The etimation of conditional quantile curves using weighted simultaneous objective function involving non-crossing constraints.

Usage

```
QRWSimul(VecX, tau, times, subj, X, y, d, kn, degree, lambda, gam)
```

Arguments

VecX	The representative values for each covariate used to estimate the desired conditional quantile curves.
tau	The quantiles of interest.
times	The vector of the time variable.
subj	The vector of subjects/individuals.
X	The covariate matrix containing 1 as its first column (including intercept in the model).
y	The response vector.
d	The order of the differencing operator for each covariate.

kn	The number of knots for each covariate.
degree	The degree of the B-spline basis function for each covariate.
lambda	The grid for the smoothing parameter to control the trade of between fidelity and penalty term (use a fine grid of lambda).
gam	The power used in estimating the smooting parameter for each covariate (e.g. gam=1 or gam=0.5).

Value

W	The weight for each subject corresponding to the length of its repeated measurement
alpha	The estimators of the coefficient vector of the basis B-splines.
hat_bt0	The baseline estimators.
hat_btk	The varying coefficient estimators.
qhat_h	The estimators of the τ_h-th conditional quantile curves.
Wtau	The weight of each order of quantile τ_h.

Note

Some warning messages are related to the function `rq.fit.sfn`.

Author(s)

Yudhie Andriyana

References

Andriyana, Y., Gijbels, I., and Verhasselt, A. P-splines quantile regression estimation in varying coefficient models. *Test* 23, 1 (2014a),153–194.

Andriyana, Y., Gijbels, I. and Verhasselt, A. (2014b). Quantile regression in varying coefficient models: non-crossingness and heteroscedasticity. *Manuscript*.

See Also

`rq.fit.sfn` `as.matrix.csr` `truncSP`

Examples

```
data(PM10)

PM10 = PM10[order(PM10$day,PM10$hour,decreasing=FALSE),]

y = PM10$PM10[1:200]
times = PM10$hour[1:200]
subj = PM10$day[1:200]
dim = length(y)
x0 = rep(1,200)
```

```
x1 = PM10$cars[1:200]
x2 = PM10$wind.speed[1:200]

X = cbind(x0, x1, x2)

VecX = c(1, max(x1), max(x2))

##########################
#### Input parameters ####
##########################
kn = c(10, 10, 10)
degree = c(3, 3, 3)
taus = seq(0.1,0.9,0.1)
lambdas = c(1,1.5,2)
d = c(1, 1, 1)
gam = 1
##########################


QRWSimul = QRWSimul(VecX=VecX, tau=taus, times=times, subj=subj, X=X, y=y,
           d=d, kn=kn, degree=degree, lambda=lambdas, gam=gam)

hat_bt0 = QRWSimul$hat_bt0
hat_btk = QRWSimul$hat_btk
qhat = QRWSimul$qhat
hat_Vt = QRWSimul$hat_Vt
Wtau = QRWSimul$Wtau


qhat1 = qhat[,1]
qhat2 = qhat[,2]
qhat3 = qhat[,3]
qhat4 = qhat[,4]
qhat5 = qhat[,5]
qhat6 = qhat[,6]
qhat7 = qhat[,7]
qhat8 = qhat[,8]
qhat9 = qhat[,9]


i = order(times, y, qhat1, qhat2, qhat3, qhat4, qhat5, qhat6, qhat7,
    qhat8, qhat9, hat_Vt);

times = times[i]; y = y[i]; qhat1 = qhat1[i]; qhat2=qhat2[i];
qhat3=qhat3[i]; qhat4=qhat4[i]; qhat5=qhat5[i]; qhat6=qhat6[i];
qhat7=qhat7[i]; qhat8=qhat8[i]; qhat9=qhat9[i];
hat_Vt = hat_Vt[i]

#Variability function V(t)

plot(hat_Vt~times, ylim=c(min(hat_Vt), max(hat_Vt)), xlab="hour",
    ylab="", type="l", lwd=2);
mtext(expression(hat(V)(t)), side=2, cex=1, line=3)
```

```
# Plot conditional quantiles estimators

ylim = range(qhat1, qhat9)
ylim = c(-4, 6)
plot(qhat1~times, col="magenta", cex=0.2, lty=5, lwd=2, type="l", ylim=ylim,
    xlab="time since infection", ylab="CD4 percentage after infection");
lines(qhat2~times, col="aquamarine4", cex=0.2, lty=4, lwd=2);
lines(qhat3~times, col="blue", cex=0.2, lty=3, lwd=3);
lines(qhat4~times, col="brown", cex=0.2, lty=2, lwd=2);
lines(qhat5~times, col="black", cex=0.2, lty=1, lwd=2);
lines(qhat6~times, col="orange", cex=0.2, lty=2, lwd=2)
lines(qhat7~times, col="darkcyan", cex=0.2, lty=3, lwd=3);
lines(qhat8~times, col="green", cex=0.2, lty=4, lwd=2);
lines(qhat9~times, col="red", cex=0.2, lty=5, lwd=3)

legend("bottom", c(expression(tau==0.9), expression(tau==0.8),
    expression(tau==0.7), expression(tau==0.6), expression(tau==0.5),
    expression(tau==0.4), expression(tau==0.3), expression(tau==0.2),
    expression(tau==0.1)), ncol=1, col=c("red","green","darkcyan",
    "orange","black","brown","blue","aquamarine4","magenta"),
    lwd=c(2,2,3,2,2,2,3,2,2), lty=c(5,4,3,2,1,2,3,4,5))
```

`simul_shapetest` *Testing the shape of a functional coefficient in the median and/or the variabilty function*

Description

Testing a functional coefficient of a covariate in the median and/or the variability function, considering the general hetersocedastic varying-coefficient model in Gijbels etal (2017a).

$$Y(t) = \sum_{k=0}^{p} \beta_k(t) X^{(k)}(t) + \gamma(t) \exp(\sum_{k=1}^{p} \theta_k(t) X^{(k)}(t))\epsilon(t).$$

Usage

```
simul_shapetest(times, subj, X, y, d, kn, degree, lambda, gam, v,
                         nr.bootstrap.samples, seed, test,omega)
```

Arguments

`times`	The vector of time variable.
`subj`	The vector of subject indicator.
`X`	The covariates, containing 1 as its first component (including intercept in the model).
`y`	The response vector.

d	The order of differencing operator for each covariate.
kn	The number of knot intervals for each covariate.
degree	The degree of B-spline basis for each covariate.
lambda	The grid of smoothing parameter to control the trade between fidelity and penalty term (use a fine grid of lambda).
gam	The power used in estimating the smooting parameter for each covariate (e.g. gam=1 or gam=0.5).
v	The covariate indicator for which the shape test is interested.
nr.bootstrap.samples	The number of bootstrap samples used.
seed	The seed for the random generator in the bootstrap resampling.
test	The requested type of testing, it consists two arguments: the first argument for median and the second for the variability function. "c" stands for constancy, "m" stands for monotonicity, and "conv" stands for convexity. insert NA to the other argument when only for median/ variability function is needed..
omega	A user defined constraint parameter for monotonicity or convexity (in Equation (7) of Gijbels etal (2017a)), chosen as large as possible.

Value

result	The testing procedures.
P	The p-values.
GR	The test statistics for the given data.
Gb	The bootstrap test statistics.

Note

Some warning messages are related to the function `rq.fit.sfn`.

Author(s)

Mohammed Abdulkerim Ibrahim

References

Andriyana, Y. and Gijbels, I. & Verhasselt, A. (2014). P-splines quantile regression estimation in varying coefficient models. *Test*, 23, 153-194.

Andriyana, Y., Gijbels, I. and Verhasselt, A. (2017). Quantile regression in varying-coefficient models: non-crossing quantile curves and heteroscedasticity. *Statistical Papers,* DOI:10.1007/s00362-016-0847-7

Gijbels, I., Ibrahim, M. A., and Verhasselt, A. (2017a). Shape testing in quantile varying coefficient models with heteroscedastic error. *Journal of Nonparametric Statistics, 29(2):391-406.*

Gijbels, I., Ibrahim, M. A., and Verhasselt, A. (2017b). Testing the heteroscedastic error structure in quantile varying coefficient models. *The Canadian Journal of Statistics,* DOI:10.1002/cjs.11346.

He, X. (1997). Quantile curves without crossing. *The American Statistician*, 51, 186-192.

See Also

rq.fit.sfn as.matrix.csr truncSP

Examples

```
data(wages)
y = wages$resp ## the hourly wage
times = wages$exper ## the duration of work experience in years
subj = wages$id ## subject indicator (individual)
dim=length(y) ## number of rows in the data = 6402
x0 = rep(1,dim) ## for intercept
### the covariates
## creating 2 dummy variables for the race covariate
wages$r1[wages$race=="black"]=1
wages$r1[wages$race!="black"]=0
wages$r2[wages$race=="hisp"]=1
wages$r2[wages$race!="hisp"]=0
x1 = wages$r1 # stands for black
x2 = wages$r2 # stands for hispanic
x3 = wages$hgc ## the highest grade completed by the indiviadual
X = cbind(x0, x1, x2, x3) ## the covariate matrix
px=ncol(X)

##########################
### Input parameters ####
#########################
lambda = 1 # we used 10^seq(-2, 1, 0.1) in Gijbels etal (2017a)
kn = rep(1,px) # used rep(5,px) in Gijbels etal (2017a)
degree = rep(2,px) # the degree of splines
d = rep(1,px)
gam=0.25
nr.bootstrap.samples=2 # used 200 in Gijbels etal (2017a)
seed=110
#########################
test1=simul_shapetest(times=times, subj=subj, X=X, y=y, d=d, kn=kn,
                     degree=degree, lambda=lambda, gam=gam, v=1,
                     nr.bootstrap.samples=nr.bootstrap.samples,seed=seed,
                     test=c("c",NA),omega=10^3)
#### Testing results
test1$result  #the testing procedures
test1$P  ## p-values
test1$R ## test statistics
```

---

test_variability *Variability Estimation and Testing*

Description

Estimating and Testing the variability function using the following hetersocedastic varying-coefficient model.

$$Y(t) = \sum_{k=0}^{p} \beta_k(t) X^{(k)}(t) + V(X(t), t)\epsilon(t)$$

where V(X(t),t) one of the six variability function in Gijbels etal (2017).

Usage

```
test_variability(times, subj, X, y, d, kn, degree, lambda, gam,tau,
                        nr.bootstrap.samples, seed, test, model)
```

Arguments

times	The vector of time variable.
subj	The vector of subjects/individuals.
X	The covariate containing 1 as its first component (including intercept in the model).
y	The response vector.
d	The order of differencing operator for each covariate.
kn	The number of knot intervals for each covariate.
degree	The degree of B-spline basis for each covariate.
lambda	The grid of smoothing parameter to control the trade between fidelity and penalty term (use a fine grid of lambda).
gam	The power used in estimating the smooting parameter for each covariate (e.g. gam=1 or gam=0.5).
tau	The quantiles of interest.
nr.bootstrap.samples	The number of bootstrap samples used.
seed	The seed for the random generator in the bootstrap resampling.
test	To request for testing the specific shape of the variability function ("Y" for test and "N" for only estimation of the parameters, the default is "Y").
model	The variability model used to estimate the quantile of errors (the default is 4, model 4).

Value

est_median	the median estimator.
hat_bt50	The median coefficients estimators.
qhat5_s2_m0	The variability (model 0) estimator.
qhat5_s2_m1	The variability (model 1) estimator.
qhat5_s2_m2	The variability (model 2) estimator.

qhat5_s2_m3	The variability (model 3) estimator.
qhat5_s2_m4	The variability (model 4) estimator.
qhat5_s2_m5	The variability (model 5) estimator.
hat_btV_0	The variability coefficients (model 0) estimators.
hat_btV_1	The variability coefficients (model 1) estimators.
hat_btV_2	The variability coefficients (model 2) estimators.
hat_btV_3	The variability coefficients (model 3) estimators.
hat_btV_4	The variability coefficients (model 4) estimators.
hat_btV_5	The variability coefficients (model 5) estimators.
C	The estimators of the tau-th quantile of the estimated residuals.
comp	The pairwise comparisons for testing the variabilty function.
P	The p-values.
GR	The test statistics for the given data.
Gb	The bootstrap test statistics.

Note

Some warning messages are related to the function `rq.fit.sfn`.

Author(s)

Mohammed Abdulkerim Ibrahim

References

Andriyana, Y. and Gijbels, I. & Verhasselt, A. (2014). P-splines quantile regression estimation in varying coefficient models. *Test*, 23, 153-194.

Andriyana, Y., Gijbels, I. and Verhasselt, A. (2017). Quantile regression in varying-coefficient models: non-crossing quantile curves and heteroscedasticity. *Statistical Papers,* DOI:10.1007/s00362-016-0847-7.

Gijbels, I., Ibrahim, M. A., and Verhasselt, A. (2017). Testing the heteroscedastic error structure in quantile varying coefficient models. *The Canadian Journal of Statistics,* DOI:10.1002/cjs.11346.

He, X. (1997). Quantile curves without crossing. *The American Statistician*, 51, 186-192.

See Also

`rq.fit.sfn` `as.matrix.csr`

Examples

```
##############################################################################
##### The real data Example in Section S3 of the supplementary material
##############################################################################
data(PM10)
PM10 = PM10[order(PM10$day,PM10$hour,decreasing=FALSE),]

y = PM10$PM10 ## the logarithm of the concentration of PM10
times = PM10$hour  ## the time in hours
subj = PM10$day  ## subject indicator (day)
dim=length(y) ## number of rows in the data = 500
x0 = rep(1,dim) ## for intercept
# the covariates
x1 = PM10$cars ## logarithm of number of cars per hour
x2 = PM10$wind.speed ## the wind speed (in meters/second)
x3 = PM10$temp ## the temperature (in degree Celsius)
X = cbind(x0, x1, x2, x3) ## the covariate matrix
px=ncol(X)

##########################
### Input parameters ####
#########################
lambda = 1 # used 10^seq(-2, 1, 0.1) in Gijbels etal (2017)
kn = rep(3,px) # used rep(10,px) in Gijbels etal (2017)
degree = rep(3,px)
d = rep(1,px)
gam=0.25
nr.bootstrap.samples= 4 # used 200 in Gijbels etal (2017)
seed=110
taus = 0.1
#########################

test1=test_variability(times=times, subj=subj, X=X, y=y, d=d, kn=kn,
                       degree=degree, lambda=lambda, gam=gam, tau=taus,
                       nr.bootstrap.samples=nr.bootstrap.samples,seed=seed,
                       test="Y",model=4)
#### Testing results
test1$comp  #the comparisons
test1$P  ## p-values
test1$GR ## test statistics

### estimation results
qhat5_s2_m4=test1$qhat5_s2_m4
qhat5_s2_m5=test1$qhat5_s2_m5
qhat5_s2_m0=test1$qhat5_s2_m0*rep(1,dim)
gamma0=test1$hat_btV_4[1:dim]
gamma1=test1$hat_btV_4[(dim+1):(dim*2)]
gamma2=test1$hat_btV_4[(dim*2+1):(dim*3)]
gamma3=test1$hat_btV_4[(dim*3+1):(dim*4)]

i = order(times, qhat5_s2_m4, qhat5_s2_m5, qhat5_s2_m0,gamma0,gamma1,
gamma2,gamma3);
```

```
times_o = times[i];  qhat5_s2_m4_o=qhat5_s2_m4[i];
qhat5_s2_m5_o=qhat5_s2_m5[i]; qhat5_s2_m0_o=qhat5_s2_m0[i]; gamma0_o=gamma0[i];
gamma1_o=gamma1[i]; gamma2_o=gamma2[i];gamma3_o=gamma3[i]

#####  variability functions plots
plot(qhat5_s2_m4_o~times_o, col="magenta", cex=0.2,
xlab="hour", ylab="estimated variability function")
lines(qhat5_s2_m5_o~times_o, col="red", cex=0.2, lty=1, lwd=2);
lines(qhat5_s2_m0_o~times_o, col="black", cex=0.2, lty=5, lwd=2);
legend("topleft", c("Model 4", "Model 5", "Model 0"), ncol=1,
        col=c("magenta","red","black"), lwd=c(1,2,2), lty=c(3,1,5))

### Plot of coefficients for variability function
plot(gamma0_o~times_o, lwd=2, type="l", xlab="hour",
ylab=expression(hat(gamma)(T)));
plot(gamma1_o~times_o, lwd=2, type="l", xlab="hour",
ylab="coefficient of logarithm of number of cars per hour");
plot(gamma2_o~times_o, lwd=2, type="l", xlab="hour",
ylab="coefficient of wind speed");
plot(gamma3_o~times_o, lwd=2, type="l", xlab="hour",
ylab="coefficient of temperature")

## Not run:
################################################################################
###############  The real data Example in Section 6 of Gijbels etal (2017)
################################################################################
data(CD4)

subj = CD4$subj ## subject indicator (a man)
dim = length(subj) ## number of rows in the data = 1817
y = CD4$CD4 ## the CD4 percentage
X0 = rep(1,dim) ## the intercept
X1 = CD4$Smooking ## the smoking status
X2 = CD4$Age ## age at HIV infection
X3 = CD4$PreCD4 ## the pre-infection CD4 percentage
times = CD4$Time ## the time in years
X = cbind(X0, X1, X2, X3) ## the covariate matrix
px=ncol(X)

lambdas = c(0.01,1,10) # used 10^seq(-2, 1, 0.1) in Gijbels etal (2017)
kn = rep(10,px) # the number of internal knots for each covariate
degree = rep(3,px) # the degree of splines
d = rep(1,px) ## The differencing order in the penalty term for each covariate
gam=0.25  ## the smooting parameter for each covariate
nr.bootstrap.samples=100 ## used 200 in Gijbels etal (2017)
seed=110 ## the seed for the random generator in the bootstrap resampling
taus = seq(0.1,0.9,0.2)

test2=test_variability(times=times, subj=subj, X=X, y=y, d=d, kn=kn,
                         degree=degree, lambda=lambdas, gam=gam,tau=taus,
                         nr.bootstrap.samples=nr.bootstrap.samples,seed=seed,
                         test="Y",model=4)
```

```
test2$comp
test2$P  ## p-values
test2$GR ## test statistics

### estimation results
  qhat5_s2_m4=test2$qhat5_s2_m4
  qhat5_s2_m5=test2$qhat5_s2_m5
  qhat5_s2_m0=test2$qhat5_s2_m0*rep(1,dim)
  gamma0=test2$hat_btV_4[1:dim]
  gamma1=test2$hat_btV_4[(dim+1):(dim*2)]
  gamma2=test2$hat_btV_4[(dim*2+1):(dim*3)]
  gamma3=test2$hat_btV_4[(dim*3+1):(dim*4)]

i = order(times, qhat5_s2_m4, qhat5_s2_m5, qhat5_s2_m0,gamma0,gamma1,
            gamma2,gamma3);
times_o = times[i];  qhat5_s2_m4_o=qhat5_s2_m4[i]; qhat5_s2_m5_o=qhat5_s2_m5[i]
qhat5_s2_m0_o=qhat5_s2_m0[i]; gamma0_o=gamma0[i]; gamma1_o=gamma1[i];
gamma2_o=gamma2[i];gamma3_o=gamma3[i]

#####  variability functions plots
plot(qhat5_s2_m4_o~times_o, col="black", cex=0.2, xlab="time since infection",
ylab="estimated variability function")
lines(qhat5_s2_m5_o~times_o, col="red", cex=0.2, lty=5, lwd=2);
lines(qhat5_s2_m0_o~times_o, col="magenta", cex=0.2, lty=1, lwd=2);
legend("topleft", c("Model 4", "Model 5", "Model 0"),
         ncol=1, col=c("black","red","magenta"),
         lwd=c(1,2,2), lty=c(3,5,1))

### Plot of coefficients for variability function
plot(gamma0_o~times_o, lwd=2, type="l", xlab="time since infection",
ylab=expression(hat(gamma)(T)));
plot(gamma1_o~times_o, lwd=2, type="l", xlab="time since infection",
ylab="coefficient of smoking status");
plot(gamma2_o~times_o, lwd=2, type="l", xlab="time since infection",
ylab="coefficient of age");
plot(gamma3_o~times_o, lwd=2, type="l", xlab="time since infection",
ylab="coefficient of pre-infection CD4")

## End(Not run)
```

---

wages *The wages dataset*

Description

This dataset is taken from the National Longitudinal Survey of Youth, collected in U.S.A. The data contain repeated measurements for 888 individuals of age 14 – 17 years old.

Usage

```
wages
```

Format

a `data.frame` with 7 variables and 6402 observations.

Details

It contains the following variables:

id the subject (individual) indicator

exper the duration of work experience in years

hgc the highest grade completed by the indiviadual

race the race of the individual: black, hispanic and white

resp the hourly wage

r1 the dummy variable for black individuals

r2 the dummy variable for hispanic individuals

References

Murnane, R. J., Willett, J. B., and Boudett, K. P. (1999). Do male dropouts benefit from obtaining a GED, postsecondary education, and training? *Evaluation Review*, 23(5):475-503.

Index