

A priedas. M0opt1 programos gradientų skaičiavimo paprogramės

Programa M0opt1 parašyta Fortran programavimo kalba. Skirta projektiniam uždaviniui (4.13)–(4.17) spręsti. Programos pagrindas Rozeno projektuojamųjų gradientų metodas. Kaip matyti iš metodo pavadinimo, vienas iš jo esminių ypatybių norint adaptuoti konkrečiam uždaviniui yra gradientų formavimas tikslo funkcijai ir apribojimams. Būtent šį funkcionalumą ir realizuoja toliau pateiktos paprogramės.

A1. Tikslo funkcijos skaičiavimo paprogramė

```
SUBROUTINE TF(mm, ieil, itask, TAKSIZE, ivirs, fsize, xm0size, AK1FI, FI, FI_G_FIT, SEVIRS, xsize,
X, XSV, ATF)

  IMPLICIT NONE

  INTEGER*4, INTENT(IN) :: mm, ieil, TAKSIZE, itask, ivirs, fsize, xsize, xm0size
  INTEGER*4 :: i, j, iBeginRow, iEndRow, iBeginCol, iEndCol, ifisize
  REAL*8, INTENT(IN) :: FI(TAKSIZE*itask, mm), X(xsize), FI_G_FIT(TAKSIZE*itask, TAKSIZE*itask),
&
  SEVIRS(mm, ivirs), AK1FI(TAKSIZE*itask, xm0size), XSV(xm0size)
  REAL*8, INTENT(OUT) :: ATF
  REAL*8 :: M0(xm0size), LAMBDA(xsize-xm0size), LAMBDA SUM((xsize-xm0size)/ivirs),
ILAMBDA(TAKSIZE*itask, TAKSIZE*itask), &
  IM0(xm0size, xm0size), BT(xsize-xm0size), ATFTMP(1)

  ATF = 0.
  ATFTMP = 0.
  BT = 0.

  ifisize = TAKSIZE*itask

  !NEZINOMUJU VEKTORIAUS X LAMBDA KOMPONENTE
  LAMBDA = 0.
  LAMBDA = X(1:xsize-xm0size)

  !NEZINOMUJU VEKTORIAUS X RIBINIŲ MOMENTŲ KOMPONENTE
  M0 = 0.
  M0 = X(xsize-xm0size+1 : xsize)

  !PJUVIŲ LAMBDA SUMA NUO KIEKVIENOS VIRŠUNES
  LAMBDA SUM = 0.
  DO i = 1, ivirs
    LAMBDA SUM = LAMBDA SUM + LAMBDA((i-1)*ifisize+1 : i*ifisize)
  END DO

  ! 1) ivirs APRIBOJIMU: [AK1FI]{M0} - [FI][G][FI]'{LAMBDA SUM} - [FI]{SE} >= 0
  !FORMUOJAMA APRIBOJIMU DALIS [AK1FI]{M0} - [FI][G][FI]'{LAMBDA SUM}
  BT(1:ifisize) = MATMUL(AK1FI, M0) - MATMUL(FI_G_FIT, LAMBDA SUM)
  !SI DALIS KOPIJUOJAMA ivirs-1 KARTU
  DO i = 2, ivirs
    BT((i-1)*ifisize+1 : i*ifisize) = BT(1 : ifisize)
  END DO
  !ATIMAMA [FI]{SE} DALIS KIEKVIENAI VIRŠUNEI
  DO i = 1, ivirs
    BT((i-1)*ifisize+1 : i*ifisize) = &
      BT((i-1)*ifisize+1 : i*ifisize) - MATMUL(FI, SEVIRS(:,i))
  END DO

  CALL GMPRD(XSV, M0, ATFTMP, 1, xm0size, 1)
  ATF = -1.*ATFTMP(1)
```

```

CALL GMPRD(LAMBDA, BT, ATFTMP, 1, xsize-xm0size, 1)
ATF = ATF - ATFTMP(1)

END SUBROUTINE

```

A2. Tikslų funkcijos gradiento skaičiavimo paprogramė

```

SUBROUTINE TFGRAD(mm, ieil, TAKSIZE, itask, ivirs, fsize, xm0size, AK1FI, FI, FI_G_FIT, SEVIRS,
xsize, X, XSV, GR)

IMPLICIT NONE

INTEGER*4, INTENT(IN) :: mm, ieil, itask, TAKSIZE, ivirs, fsize, xsize, xm0size
INTEGER*4 :: i, j, iBeginRow, iEndRow, iBeginCol, iEndCol, ifsize
REAL*8, INTENT(IN) :: FI(TAKSIZE*itask, mm), X(xsize), FI_G_FIT(TAKSIZE*itask, TAKSIZE*itask),
&
SEVIRS(mm, ivirs), AK1FI(TAKSIZE*itask, xm0size), XSV(xm0size)
REAL*8, INTENT(OUT) :: GR(xsize)
REAL*8 :: M0(xm0size), LAMBDA(xsize-xm0size), LAMBDA SUM((xsize-xm0size)/ivirs),
ILAMBDA(TAKSIZE*itask, TAKSIZE*itask), &
IM0(xm0size, xm0size), BT(xsize-xm0size)

GR = 0.

ifsize = TAKSIZE*itask

!NEZINOMUJU VEKTORIAUS X LAMBDA KOMPONENTE
LAMBDA = 0.
LAMBDA = X(1:xsize-xm0size)

!NEZINOMUJU VEKTORIAUS X RIBINIŲ MOMENTŲ KOMPONENTE
M0 = 0.
M0 = X(xsize-xm0size+1 : xsize)

!PJUVIŲ LAMBDA SUMA NUO KIEKVIENTOS VIRŠUNES
LAMBDA SUM = 0.
DO i = 1, ivirs
    LAMBDA SUM = LAMBDA SUM + LAMBDA((i-1)*ifsize+1 : i*ifsize)
END DO

! FORMUOJAMA GRADIENTO DALIS -[AK1FI]{M0} + 2.*[FI][G][FI]'{LAMBDA SUM}
GR(1:ifsize) = -1.*MATMUL(AK1FI, M0) + 2.*MATMUL(FI_G_FIT, LAMBDA SUM)
!SI DALIS KOPIJUOJAMA ivirs-1 KARTU
DO i = 2, ivirs
    GR((i-1)*ifsize+1 : i*ifsize) = GR(1 : ifsize)
END DO

! PRIDEDAMA [FI][ALFAVIRS]{SE} DALIS KIEKVIENTAI VIRŠUNEI
DO i = 1, ivirs
    GR((i-1)*ifsize+1 : i*ifsize) = &
        GR((i-1)*ifsize+1 : i*ifsize) + MATMUL(FI, SEVIRS(:, i))
END DO
GR(xsize-xm0size+1 : xsize) = -1.*XSV - MATMUL(TRANSPOSE(AK1FI), LAMBDA SUM)

END SUBROUTINE

```

A3. Apribojimų ir jų gradientų skaičiavimo paprogramė

```

SUBROUTINE PUSREM(mm, ieil, itask, TAKSIZE, ivirs, fsize, xm0size, M0RIB, AK1FI, m0daug, udaug,
URRIB, FI, FI_G_FIT, H_FIT, SEVIRS, UEMAXMIN, xsize, X, btsize, BT, isv, Pl)

IMPLICIT NONE

INTEGER*4, INTENT(IN) :: mm, ieil, itask, TAKSIZE, ivirs, fsize, xsize, btsize, isv, xm0size
INTEGER*4 :: i, j, iBeginRow, iEndRow, iBeginCol, iEndCol, ifsize
REAL*8, INTENT(IN) :: FI(TAKSIZE*itask, mm), X(xsize), FI_G_FIT(TAKSIZE*itask, TAKSIZE*itask),
H_FIT(ieil, TAKSIZE*itask), &
SEVIRS(mm, ivirs), M0RIB(xm0size*2), URRIB(ieil*2), AK1FI(TAKSIZE*itask, xm0size), m0daug,
udaug, UEMAXMIN(ieil*2)
REAL*8, INTENT(OUT) :: BT(btsize), Pl(btsize, xsize)
REAL*8 :: M0(xm0size), LAMBDA(xsize-xm0size), LAMBDA SUM((xsize-xm0size)/ivirs),
ILAMBDA(TAKSIZE*itask, TAKSIZE*itask), &

```

```

IM0(xm0size, xm0size)
BT = 0.
ifisize = TAKSIZE*itask

!NEZINOMUJU VEKTORIAUS X LAMBDA KOMPONENTE
LAMBDA = 0.
LAMBDA = X(1:xsize-xm0size)

!NEZINOMUJU VEKTORIAUS X RIBINIŲ MOMENTŲ KOMPONENTE
M0 = 0.
M0 = X(xsize-xm0size+1 : xsize)

!PJUVIŲ LAMBDA SUMA NUO KIEKVIENOS VIRŠUNES
LAMBDAΣ = 0.
DO i = 1, ivirs
    LAMBDAΣ = LAMBDAΣ + LAMBDA((i-1)*ifisize+1 : i*ifisize)
END DO

! 1) ivirs APRIBOJIMU: [AK1FI]{M0} - [FI][G][FI]'{LAMBDAΣ} - [FI]{SE} >= 0
!FORMUOJAMA APRIBOJIMU DALIS [AK1FI]{M0} - [FI][G][FI]'{LAMBDAΣ}
BT(1:ifisize) = MATMUL(AK1FI, M0)/m0daug - MATMUL(FI_G_FIT, LAMBDAΣ)/m0daug
!SI DALIS KOPIJUOJAMA ivirs-1 KARTU
DO i = 2, ivirs
    BT((i-1)*ifisize+1 : i*ifisize) = BT(1 : ifisize)
END DO
!ATIMAMA [FI]{SE} DALIS KIEKVIENAI VIRŠUNEI
DO i = 1, ivirs
    BT((i-1)*ifisize+1 : i*ifisize) = &
    BT((i-1)*ifisize+1 : i*ifisize) - MATMUL(FI, SEVIRS(:,i)/m0daug)
END DO

! 2) LAMBDA >= 0
iBeginRow = ifisize*ivirs+1
iEndRow = iBeginRow-1 + xsize-xm0size
BT(iBeginRow : iEndRow) = LAMBDA

! 3) MOMAX - M0 >= 0
iBeginRow = iEndRow+1
iEndRow = iBeginRow-1 + xm0size
BT(iBeginRow : iEndRow) = MORIB(1:xm0size) - M0

! 4) M0 - MOMIN >= 0
iBeginRow = iEndRow+1
iEndRow = iBeginRow-1 + xm0size
BT(iBeginRow : iEndRow) = M0 - MORIB(xm0size+1:xm0size*2)

! 5) URMAX - [H][FI]'{LAMBDAΣ} - UEMAX >= 0
iBeginRow = iEndRow+1
iEndRow = iBeginRow-1 + ieil
BT(iBeginRow : iEndRow) = udaug*(URRIB(1:ieil) - MATMUL(H_FIT, LAMBDAΣ) - UEMAXMIN(1:ieil))

! 6) UEMIN + [H][FI]'{LAMBDAΣ} - URMIN >= 0
iBeginRow = iEndRow+1
iEndRow = iBeginRow-1 + ieil
BT(iBeginRow : iEndRow) = udaug*(UEMAXMIN(ieil+1:ieil*2) + MATMUL(H_FIT, LAMBDAΣ) -
URRIB(ieil+1:ieil*2))

IF(isv /= 0) THEN
    P1 = 0.
    ILAMBDA = 0.

    ! 1)
    DO i = 1, ivirs*ifisize, ifisize
        DO j = 1, ivirs*ifisize, ifisize
            P1(i:i+ifisize-1, j:j+ifisize-1) = -1.*FI_G_FIT
        END DO
    END DO

    iBeginCol = ifisize*ivirs+1
    iEndCol = iBeginCol-1 + xm0size
    j = 1
    DO i = 1, ivirs*ifisize, ifisize
        P1(i:i+ifisize-1, iBeginCol:iEndCol) = AK1FI
        j = j+1
    END DO

    ! 2)

```

```

FORALL(i=1:ifisize, j=1:ifisize, (i.EQ.j)) ILAMBDA(i, j) = 1.

DO i = ivirs*ifisize+1, (ivirs*ifisize + ivirs*ifisize), ifisize
  DO j = 1, ivirs*ifisize, ifisize
    IF((i-ivirs*ifisize)==j)THEN
      P1(i:i+ifisize-1, j:j+ifisize-1) = ILAMBDA
    END IF
  END DO
END DO

! 3)
IM0 = 0.
FORALL(i=1:xm0size, j=1:xm0size, (i.EQ.j)) IM0(i, j) = 1.

iBeginRow = ivirs*ifisize + ivirs*ifisize + 1
iEndRow = iBeginRow-1 + xm0size
P1(iBeginRow:iEndRow, iBeginCol:iEndCol) = -1.*IM0

! 4)
iBeginRow = iEndRow+1
iEndRow = iBeginRow-1 + xm0size
P1(iBeginRow:iEndRow, iBeginCol:iEndCol) = IM0

! 5)
iBeginRow = iEndRow+1
iEndRow = iBeginRow-1 + ie1l

DO j = 1, ivirs*ifisize, ifisize
  P1(iBeginRow:iEndRow, j:j+ifisize-1) = -1.*udaug*H_FIT
END DO

! 6)
iBeginRow = iEndRow+1
iEndRow = iBeginRow-1 + ie1l

DO j = 1, ivirs*ifisize, ifisize
  P1(iBeginRow:iEndRow, j:j+ifisize-1) = udaug*H_FIT
END DO

END IF
END SUBROUTINE

```

B priedas. MaxFopt1 programos gradientų skaičiavimo paprogramės

Programa MaxFopt1 parašyta Fortran programavimo kalba. Skirta patikrinamajam uždaviniui (4.18)–(4.22) spręsti. Programos pagrindas Rozeno projektuojamųjų gradientų metodas. Analogiškai kaip ir A priede, toliau pateikiamos tikslo funkcijos bei apribojimų reikšmių, bei gradientų formavimo paprogramės.

B1. Tikslo funkcijos skaičiavimo paprogramė

SUBROUTINE TF(mm, ieil, itask, TAKSIZE, ivirs, fsize, VERSION_NR, SRIB, FRIB, FI, FI_G_FIT, MNKFG, ALFAFVIRS, ALFA, FSUPFINFKONF, FKOMB, SEPC, xsize, X, TFSV, ATF)

```

IMPLICIT NONE

INTEGER*4, INTENT(IN) :: mm, ieil, itask, ivirs, fsize, xsize, TAKSIZE
INTEGER*4 :: i, j, iBeginRow, iEndRow, iBeginCol, iEndCol, ifisize
REAL*8, INTENT(IN) :: SRIB(mm), FI(TAKSIZE*itask, mm), X(xsize), FI_G_FIT(TAKSIZE*itask,
TAKSIZE*itask), &
ALFAFVIRS(mm, fsize*2, ivirs), FRIB(fsize*2), TFSV(FSIZE*2), ALFA(MM, IEIL),
FSUPFINFKONF(fsize*2, IEIL), SEPC(mm), VERSION_NR
INTEGER*4, INTENT(IN) :: MNKFG(itask, 2), FKOMB(IVIRS, FSIZE)
REAL*8, INTENT(OUT) :: ATF
REAL*8 :: M0(TAKSIZE*itask), LAMBDA(xsize-fsize*2), LAMBDA SUM((xsize-fsize*2)/ivirs),
FSUPFINF(fsize*2), ILAMBDA(TAKSIZE*itask, TAKSIZE*itask), &
IFSUPFINF(fsize*2, fsize*2), ATFTMP(1), BT(xsize-fsize*2), Se(mm)

ATF = 0.
ATFTMP = 0.
BT = 0.

ifisize = TAKSIZE*itask

!PJUVIU RIBINIŲ MOMENTŲ REIKSMES
M0 = 0.
j = 1
DO i = 1, ifisize, TAKSIZE
    M0(i) = SRIB(MNKFG(j, 1))
    M0(i+1) = M0(i)
    M0(i+2) = M0(i)
    M0(i+3) = M0(i)
    IF(TAKSIZE == 6) THEN
        M0(i+4) = M0(i)
        M0(i+5) = M0(i)
    END IF
    j = j+1
END DO

!NEZINOMUJŲ VEKTORIAUS X LAMBDA KOMPONENTE
LAMBDA = 0.
LAMBDA = X(1:xsize-fsize*2)

!NEZINOMUJŲ VEKTORIAUS X APKROVOS RIBŲ KOMPONENTE
FSUPFINF = 0.
FSUPFINF = X(xsize-fsize*2+1 : xsize)

DO i = 1, fsize*2
    IF(i<=fsize) THEN
        ATFTMP(1) = ATFTMP(1) + TFSV(i)*FSUPFINF(i)
    ELSE
        ATFTMP(1) = ATFTMP(1) - TFSV(i)*FSUPFINF(i)
    END IF
END DO

```

```

!PJUVIU LAMBDA SUMA NUO KIEKVIENOS VIRSUNES
LAMBDA SUM = 0.
DO i = 1, ivirs
    LAMBDA SUM = LAMBDA SUM + LAMBDA((i-1)*ifsize+1 : i*ifsize)
END DO

! 1) ivirs APRIBOJIMU: {M0} - [FI][G][FI]'{LAMBDA SUM} - [FI][ALFAVIRS]{SE} >= 0
!FORMUOJAMA APRIBOJIMU DALIS {M0} - [FI][G][FI]'{LAMBDA SUM}
BT(1:ifsize) = M0 - MATMUL(FI G FIT, LAMBDA SUM)
!SI DALIS KOPIJUOJAMA ivirs-1 KARTU
DO i = 2, ivirs
    BT((i-1)*ifsize+1 : i*ifsize) = BT(1 : ifsize)
END DO
!ATIMAMA [FI][ALFAVIRS]{SE} DALIS KIEKVIENAI VIRSUNEI
DO i = 1, ivirs
    Se = 0.
    IF(VERSION NR == 1.0) THEN
        Se = MATMUL(ALFAFVIRS(1:mm, 1:fsize*2, i), FSUPFINF)
    ELSE
        DO j = 1, FSIZE
            IF(FKOMB(i, j) == 1) THEN
                Se = Se + MATMUL(ALFA, (FSUPFINFKONF(j, :) * FSUPFINF(j)))
            ELSE
                Se = Se + MATMUL(ALFA, (FSUPFINFKONF(fsize+j, :) * FSUPFINF(fsize+j)))
            END IF
        END DO
        Se = Se + SEPC
    END IF
    BT((i-1)*ifsize+1 : i*ifsize) = &
        BT((i-1)*ifsize+1 : i*ifsize) - MATMUL(FI, Se)
END DO

ATF = ATFTMP(1)

CALL GMPRD(LAMBDA, BT, ATFTMP, 1, xsize-fsize*2, 1)

ATF = ATF - ATFTMP(1)

END SUBROUTINE

```

B2. Tikslų funkcijos gradiento skaičiavimo paprogramė

SUBROUTINE TFGRAD(mm, iel, itask, TAKSIZE, ivirs, fsize, VERSION NR, SRIB, FRIB, FI, FI_G_FIT, MNKFG, ALFAFVIRS, ALFA, FSUPFINFKONF, FKOMB, SEPC, xsize, X, TFSV, GR)

```

IMPLICIT NONE

INTEGER*4, INTENT(IN) :: mm, iel, itask, ivirs, fsize, xsize, TAKSIZE
INTEGER*4 :: i, j, iBeginRow, iEndRow, iBeginCol, iEndCol, ifsize
REAL*8, INTENT(IN) :: SRIB(mm), FI(TAKSIZE*itask, mm), X(xsize), FI_G_FIT(TAKSIZE*itask,
TAKSIZE*itask), &
    ALFAFVIRS(mm, fsize*2, ivirs), FRIB(fsize*2), TFSV(FSIZE*2), ALFA(MM, IEIL),
FSUPFINFKONF(fsize*2, IEIL), SEPC(mm), VERSION NR
INTEGER*4, INTENT(IN) :: MNKFG(itask, 2), FKOMB(IVIRS, FSIZE)
REAL*8, INTENT(OUT) :: GR(xsize)
REAL*8 :: M0(TAKSIZE*itask), LAMBDA(xsize-fsize*2), LAMBDA SUM((xsize-fsize*2)/ivirs),
FSUPFINF(fsize*2), ILAMBDA(TAKSIZE*itask, TAKSIZE*itask), &
    FSUPFINF(fsize*2, fsize*2), Se(mm), FI1(TAKSIZE*itask), TMP(1)

GR = 0.

ifsize = TAKSIZE*itask

!PJUVIU RIBINIŲ MOMENTŲ REIKSMES
M0 = 0.
j = 1
DO i = 1, ifsize, TAKSIZE
    M0(i) = SRIB(MNKFG(j, 1))
    M0(i+1) = M0(i)
    M0(i+2) = M0(i)
    M0(i+3) = M0(i)
    IF(TAKSIZE == 6) THEN

```

```

        M0(i+4) = M0(i)
        M0(i+5) = M0(i)
    END IF
    j = j+1
END DO

!NEZINOMUJU VEKTORIAUS X LAMBDA KOMPONENTE
LAMBDA = 0.
LAMBDA = X(1:xsize-fsize*2)

!NEZINOMUJU VEKTORIAUS X APKROVOS RIBU KOMPONENTE
FSUPFINF = 0.
FSUPFINF = X(xsize-fsize*2+1 : xsize)

!PJUVIU LAMBDA SUMA NUO KIEKVIENOS VIRSUNES
LAMBDA SUM = 0.
DO i = 1, ivirs
    LAMBDA SUM = LAMBDA SUM + LAMBDA((i-1)*ifsize+1 : i*ifsize)
END DO

!FORMUOJAMA GRADIENTO DALIS -(M0) + 2*[FI][G][FI]'{LAMBDA SUM}
GR(1:ifsize) = -M0 + 2.*MATMUL(FI_G_FIT, LAMBDA SUM)
!SI DALIS KOPIJUOJAMA ivirs-1 KARTU
DO i = 2, ivirs
    GR((i-1)*ifsize+1 : i*ifsize) = GR(1 : ifsize)
END DO
!PRIEDAMA [FI][ALFAVIRS]{SE} DALIS KIEKVIENAI VIRSUNEI
DO i = 1, ivirs
    Se = 0.
    IF(VERSION_NR == 1.0) THEN
        Se = MATMUL(ALFAFVIRS(1:mm, 1:fsize*2, i), FSUPFINF)
    ELSE
        DO j = 1, FSIZE
            IF(FKOMB(i, j) == 1) THEN
                Se = Se + MATMUL(ALFA, (FSUPFINFKONF(j, :)*FSUPFINF(j)))
            ELSE
                Se = Se + MATMUL(ALFA, (FSUPFINFKONF(fsize+j, :)*FSUPFINF(fsize+j)))
            END IF
        END DO
        Se = Se + SEPC
    END IF
    GR((i-1)*ifsize+1 : i*ifsize) = &
        GR((i-1)*ifsize+1 : i*ifsize) + MATMUL(FI, Se)
END DO

GR(xsize-fsize*2+1 : xsize-fsize) = TFSV(1:fsize)
GR(xsize-fsize+1 : xsize) = -1. * TFSV(fsize+1:fsize*2)
IF(VERSION_NR == 1.0) THEN
    DO i = 1, ivirs
        GR(xsize-fsize*2+1 : xsize) = &
            GR(xsize-fsize*2+1 : xsize) + MATMUL(LAMBDA((i-1)*ifsize+1 : i*ifsize),
            MATMUL(FI, ALFAFVIRS(1:mm, 1:fsize*2, i)))
    END DO
ELSE
    DO i = 1, ivirs
        DO j = 1, FSIZE
            IF(FKOMB(i, j) == 1) THEN
                FI1 = MATMUL(FI, MATMUL(ALFA, FSUPFINFKONF(j, :)))
                CALL GMPRD(LAMBDA((i-1)*ifsize+1 : i*ifsize), FI1, TMP, 1, ifsize, 1)
                GR(xsize-fsize*2+j) = GR(xsize-fsize*2+j) + TMP(1)
            ELSE
                FI1 = MATMUL(FI, MATMUL(ALFA, FSUPFINFKONF(fsize+j, :)))
                CALL GMPRD(LAMBDA((i-1)*ifsize+1 : i*ifsize), FI1, TMP, 1, ifsize, 1)
                GR(xsize-fsize*2+FSIZE+j) = GR(xsize-fsize*2+FSIZE+j) + TMP(1)
            END IF
        END DO
    END DO
END IF
END SUBROUTINE

```

B3. Apribojimų ir jų gradientų skaičiavimo paprogramė

```
SUBROUTINE PUSREM(mm, ieil, itask, TAKSIZE, ivirs, fsize, SRIB, FRIB, fdaug, udaug, VERSION_NR,
BSUMU, DELTA, URRIB, FI, G, FI_G FIT, H FIT, MNKFG, ALFAFVIRS, ALFA, BETA, FSUPFINFKONF, FKOMB,
SEPC, UEPC, xsize, X, btsize, BT, isv, P1, MISES)
```

```
IMPLICIT NONE
```

```
INTEGER*4, INTENT(IN) :: mm, ieil, itask, ivirs, fsize, xsize, btsize, isv, TAKSIZE, BSUMU,
MISES
```

```
INTEGER*4 :: i, j, k, iBeginRow, iEndRow, iBeginCol, iEndCol, ifisize
REAL*8, INTENT(IN) :: SRIB(mm), FI(TAKSIZE*itask, mm), G(mm, mm), X(xsize),
FI_G FIT(TAKSIZE*itask, TAKSIZE*itask), H FIT(ieil, TAKSIZE*itask), &
ALFAFVIRS(mm, fsize*2, ivirs), FRIB(fsize*2), URRIB(ieil*2), fdaug, udaug, ALFA(MM, IEIL),
FSUPFINFKONF(fsize*2, IEIL), SEPC(mm), UEPC(ieil), BETA(IEIL, IEIL), VERSION_NR, DELTA
INTEGER*4, INTENT(IN) :: MNKFG(itask, 2), FKOMB(IVIRS, FSIZE)
REAL*8, INTENT(OUT) :: BT(btsize), P1(btsize, xsize)
REAL*8 :: M0(TAKSIZE*itask), LAMBDA(xsize-fsize*2), LAMBDA SUM((xsize-fsize*2)/ivirs),
FSUPFINF(fsize*2), ILAMBDA(TAKSIZE*itask, TAKSIZE*itask), &
IFSUPFINF(fsize*2, fsize*2), Sr(mm), Se(mm), Ssum(mm), UEVIRS(IEIL, IVIRS), Ue(ieil), SEVIRS(mm,
ivirs), UR(IEIL), UEMAXMIN(2*ieil), TMP(1)
```

```
BT = 0.
Sr = 0.
Se = 0.
UEVIRS = 0.
SEVIRS = 0.
```

```
ifisize = TAKSIZE*itask
```

```
!PJUVIU RIBINIŲ MOMENTŲ REIKSMES
```

```
M0 = 0.
j = 1
DO i = 1, ifisize, TAKSIZE
    M0(i) = SRIB(MNKFG(j, 1))
    M0(i+1) = M0(i)
    M0(i+2) = M0(i)
    M0(i+3) = M0(i)
    IF(TAKSIZE == 6) THEN
        M0(i+4) = M0(i)
        M0(i+5) = M0(i)
    END IF
    j = j+1
END DO
```

```
!NEZINOMUJŲ VEKTORIAUS X LAMBDA KOMPONENTE
```

```
LAMBDA = 0.
LAMBDA = X(1:xsize-fsize*2)
```

```
!NEZINOMUJŲ VEKTORIAUS X APKROVOS RIBŲ KOMPONENTE
```

```
FSUPFINF = 0.
FSUPFINF = X(xsize-fsize*2+1 : xsize)
```

```
!PJUVIU LAMBDA SUMA NUO KIEKVIENTOS VIRSUNES
```

```
LAMBDA SUM = 0.
DO i = 1, ivirs
    LAMBDA SUM = LAMBDA SUM + LAMBDA((i-1)*ifisize+1 : i*ifisize)
END DO
```

```
! 1) ivirs APRIBOJIMU: {M0} - [FI][G][FI]'{LAMBDA SUM} - [FI][ALFAVIRS]{SE} >= 0
!FORMUOJAMA APRIBOJIMU DALIS {M0} - [FI][G][FI]'{LAMBDA SUM}
Sr = MATMUL(MATMUL(G, TRANSPOSE(FI)), LAMBDA SUM);
Sr = Sr/fdaug
```

```
BT(1:ifisize) = M0/fdaug - MATMUL(FI, Sr)
```

```
!SI DALIS KOPIJUOJAMA ivirs-1 KARTU
```

```
DO i = 2, ivirs
    BT((i-1)*ifisize+1 : i*ifisize) = BT(1 : ifisize)
END DO
```

```
!ATIMAMA [FI][ALFAVIRS]{SE} DALIS KIEKVIENTAI VIRSUNEI
```

```
DO i = 1, ivirs
    Se = 0.
    IF(VERSION_NR == 1.0) THEN
        Se = MATMUL(ALFAFVIRS(1:mm, 1:fsize*2, i), FSUPFINF)
        Se = Se/fdaug
```



```

ELSE
  DO j = 1, FSIZE
    IF(FKOMB(i, j) == 1) THEN
      Se = Se + MATMUL(ALFA, (FSUPFINFKONF(j, :)*FSUPFINF(j)))
    ELSE
      Se = Se + MATMUL(ALFA, (FSUPFINFKONF(fsize+j, :)*FSUPFINF(fsize+j)))
    END IF
  END DO
  Se = Se + SEPC
  Se = Se/fdaug
  SEVIRS(:, i) = Se
END IF
BT((i-1)*ifisize+1 : i*ifisize) = &
  BT((i-1)*ifisize+1 : i*ifisize) - MATMUL(FI, Se)
Ssum = Se + Sr

IF(isv == -1 .AND. VERSION_NR == 1.0) THEN
  WRITE(8, *) ' '
  WRITE(8, *) ' ***** VIRSUNES ', i, ' SKAICIAVIMO REZULTATAI. PRADZIA
*****'
  WRITE(8, *) ' '
  CALL TABELR1EXPL(M0, 1, ifisize, 'RIBINIŲ MOMENTŲ VEKTORIAUS', 8)
  CALL TABELR1EXPL(Sr, 1, mm, 'LIEKAMŲJŲ IRAZŲ VEKTORIAUS', 8)
  CALL TABELR1EXPL(Se, 1, mm, 'TAMPRIŲ IRAZŲ VEKTORIAUS', 8)
  CALL TABELR1EXPL(Ssum, 1, mm, 'SUMINIŲ IRAZŲ VEKTORIAUS', 8)
  CALL TABELR1EXPL(MATMUL(FI, Sr), 1, ifisize, 'FI*Sr', 8)
  CALL TABELR1EXPL(MATMUL(FI, Se), 1, ifisize, 'FI*Se', 8)
  CALL TABELR1EXPL(BT((i-1)*ifisize+1 : i*ifisize), 1, ifisize, 'BT', 8)
  WRITE(8, *) ' '
  WRITE(8, *) ' ***** VIRSUNES ', i, ' SKAICIAVIMO REZULTATAI. PABAIGA
*****'
  WRITE(8, *) ' '
END IF

END DO

IF(VERSION_NR > 1.0 .AND. isv == -1) THEN
  CALL TABELR1EXPL(SEVIRS, MM, IVIRS, 'SEVIRS(MM, IVIRS) + PC(MM) - TAMPRIŲ IRAZŲ MATRICA
VIRSUNEMS', 8)
END IF

! 2) LAMBDA >= 0
iBeginRow = ifisize*ivirs+1
iEndRow = iBeginRow-1 + xsize-fsize*2
BT(iBeginRow : iEndRow) = LAMBDA

! 3) FSUP >= 0
iBeginRow = iEndRow+1
iEndRow = iBeginRow-1 + fsize
BT(iBeginRow : iEndRow) = FSUPFINF(1:fsize)

! 4) -FINF >= 0
iBeginRow = iEndRow+1
iEndRow = iBeginRow-1 + fsize
BT(iBeginRow : iEndRow) = -1.*FSUPFINF(fsize+1:fsize*2)

! 5) FSUPMAX - FSUP >= 0
iBeginRow = iEndRow+1
iEndRow = iBeginRow-1 + fsize
BT(iBeginRow : iEndRow) = FRIB(1:fsize) - FSUPFINF(1:fsize)

! 6) FINF - FINFMIN >= 0
iBeginRow = iEndRow+1
iEndRow = iBeginRow-1 + fsize
BT(iBeginRow : iEndRow) = FSUPFINF(fsize+1:fsize*2) - FRIB(fsize+1:fsize*2)

DO i = 1, ivirs
  Ue = 0.
  IF(VERSION_NR > 1.0) THEN
    DO j = 1, FSIZE
      IF(FKOMB(i, j) == 1) THEN
        Ue = Ue + MATMUL(BETA, (FSUPFINFKONF(j, :)*FSUPFINF(j)))
      ELSE
        Ue = Ue + MATMUL(BETA, (FSUPFINFKONF(fsize+j, :)*FSUPFINF(fsize+j)))
      END IF
    END DO
    Ue = Ue + UEPC
  
```

```

        UEVIRS(:, i) = Ue
    END IF
END DO

IF(VERSION_NR > 1.0 .AND. isv == -1) THEN
    CALL TABELR1EXPL(UEVIRS, IEIL, IVIRS, 'UEVIRS(IEIL, IVIRS) + UEPC(IEIL) - TAMPRIU POSLINKIU
MATRICA VIRSUNEMS', 8)
    UR = MATMUL(H_FIT, LAMBDASUM)
    IF(BSUMU /= 0) THEN
        UEMAXMIN(1:IEIL)=MAXVAL(UEVIRS, DIM = 2) !, MASK = UEVIRS.GE.0.0
        UEMAXMIN(IEIL+1:IEIL*2)=MINVAL(UEVIRS, DIM = 2)!, MASK = UEVIRS.LE.0.0
        CALL TABELR1EXPL(UEMAXMIN, 1, IEIL*2, 'UEMAXMIN(IEIL*2) - MAKSIMALIU IR MINIMALIU
TAMPRIU POSLINKIU REIKSMIU VEKTORIUS', 8)
    ELSE
        UEMAXMIN(1:IEIL)=0
        UEMAXMIN(IEIL+1:IEIL*2)=0
    END IF

    DO I=1,IEIL
        TMP(1) = (URRIB(I)-UR(I)-UEMAXMIN(I))*UDAUG
        IF(ABS(TMP(1)) <= DELTA) THEN
            WRITE(8, *) 'SUMINIS POSLINKIS NR =', I, ' PASIEKE VIRSUTINE RIBA'
        END IF

        TMP(1) = (UR(I)+UEMAXMIN(IEIL+I)-URRIB(IEIL+I))*UDAUG
        IF(ABS(TMP(1)) <= DELTA) THEN
            WRITE(8, *) 'SUMINIS POSLINKIS NR =', I, ' PASIEKE APATINE RIBA'
        END IF
    END DO
END IF

IF(BSUMU == 0) THEN
    ! 7) URMAL - [H][FI]'{LAMBDASUM} >= 0
    iBeginRow = iEndRow+1
    iEndRow = iBeginRow-1 + ieil
    BT(iBeginRow : iEndRow) = udaug*(URRIB(1:ieil) - MATMUL(H_FIT, LAMBDASUM))

    ! 8) [H][FI]'{LAMBDASUM} - URMAL >= 0
    iBeginRow = iEndRow+1
    iEndRow = iBeginRow-1 + ieil
    BT(iBeginRow : iEndRow) = udaug*(MATMUL(H_FIT, LAMBDASUM) - URRIB(ieil+1:ieil*2))
ELSE
    ! 7) URMAL - [H][FI]'{LAMBDASUM} - UEPC >= 0
    DO i = 1, ivirs
        iBeginRow = iEndRow+1
        iEndRow = iBeginRow-1 + ieil
        BT(iBeginRow : iEndRow) = udaug*(URRIB(1:ieil) - MATMUL(H_FIT, LAMBDASUM) - UEVIRS(:,i))
    END DO

    ! 8) UEPC + [H][FI]'{LAMBDASUM} - URMAL >= 0
    DO i = 1, ivirs
        iBeginRow = iEndRow+1
        iEndRow = iBeginRow-1 + ieil
        BT(iBeginRow : iEndRow) = udaug*(UEVIRS(:,i) + MATMUL(H_FIT, LAMBDASUM) -
URRIB(ieil+1:ieil*2))
    END DO
END IF

IF(isv == 5) THEN
    PI = 0.
    ILAMBDA = 0.

    ! 1)
    DO i = 1, ivirs*ifisize, ifisize
        DO j = 1, ivirs*ifisize, ifisize
            P1(i:i+ifisize-1, j:j+ifisize-1) = -1.*PI_G_FIT
        END DO
    END DO

    iBeginCol = ifisize*ivirs+1
    iEndCol = iBeginCol-1 + fsize*2
    IF(VERSION_NR == 1.0) THEN
        j = 1
        DO i = 1, ivirs*ifisize, ifisize
            P1(i:i+ifisize-1, iBeginCol:iEndCol) = -1.*MATMUL(FI, ALFAFVIRS(1:mm, 1:fsize*2, j))
            j = j+1
        END DO
    END IF

```

```

        END DO
    ELSE
        j = 1
        DO i = 1, ivirs*ifisize, ifisize
            DO k = 1, FSIZE
                IF(FKOMB(j, k) == 1) THEN
                    P1(i:i+ifisize-1, iBeginCol-1+k) = -1.*MATMUL(FI, MATMUL(ALFA,
FSUPFINFKONF(k, :)))
                ELSE
                    P1(i:i+ifisize-1, iBeginCol-1+FSIZE+k) = -1.*MATMUL(FI, MATMUL(ALFA,
FSUPFINFKONF(FSIZE+k, :)))
                END IF
            END DO
            j = j+1
        END DO
    END IF

! 2)
FORALL(i=1:ifisize, j=1:ifisize, (i.EQ.j)) ILAMBDA(i, j) = 1.

DO i = ivirs*ifisize+1, (ivirs*ifisize + ivirs*ifisize), ifisize
    DO j = 1, ivirs*ifisize, ifisize
        IF((i-ivirs*ifisize)==j) THEN
            P1(i:i+ifisize-1, j:j+ifisize-1) = ILAMBDA
        END IF
    END DO
END DO

1.
! 3) 4)
IFSUPFINF = 0.
FORALL(i=1:fsize, j=1:fsize, (i.EQ.j)) IFSUPFINF(i, j) = 1.
FORALL(i=fsize+1:fsize*2, j=fsize+1:fsize*2, (i.EQ.j)) IFSUPFINF(i, j) = -1.

iBeginRow = ivirs*ifisize + ivirs*ifisize + 1
iEndRow = iBeginRow-1 + fsize*2

P1(iBeginRow:iEndRow, iBeginCol:iEndCol) = IFSUPFINF

! 5) 6)
iBeginRow = iEndRow+1
iEndRow = iBeginRow-1 + fsize*2

P1(iBeginRow:iEndRow, iBeginCol:iEndCol) = -1.*IFSUPFINF

! 7)
IF(BSUMU == 0) THEN
    iBeginRow = iEndRow+1
    iEndRow = iBeginRow-1 + ieil
    DO j = 1, ivirs*ifisize, ifisize
        P1(iBeginRow:iEndRow, j:j+ifisize-1) = -1.*udaug*H_FIT
    END DO
ELSE
    DO i = 1, ivirs
        iBeginRow = iEndRow+1
        iEndRow = iBeginRow-1 + ieil

        DO j = 1, ivirs*ifisize, ifisize
            P1(iBeginRow:iEndRow, j:j+ifisize-1) = -1.*udaug*H_FIT
        END DO

        DO j = 1, FSIZE
            IF(FKOMB(i, j) == 1) THEN
                P1(iBeginRow:iEndRow, ivirs*ifisize+j) = -1.*MATMUL(BETA, (FSUPFINFKONF(j,
:)))
            ELSE
                P1(iBeginRow:iEndRow, ivirs*ifisize+FSIZE+j) = -1.*MATMUL(BETA,
(FSUPFINFKONF(FSIZE+j, :)))
            END IF
        END DO
    END DO
END IF

! 8)
IF(BSUMU == 0) THEN
    iBeginRow = iEndRow+1
    iEndRow = iBeginRow-1 + ieil
    DO j = 1, ivirs*ifisize, ifisize

```

```

        P1(iBeginRow:iEndRow, j:j+ifisize-1) = uaug*H_FIT
    END DO
ELSE
    DO i = 1, ivirs
        iBeginRow = iEndRow+1
        iEndRow = iBeginRow-1 + ieil

        DO j = 1, ivirs*ifisize, ifisize
            P1(iBeginRow:iEndRow, j:j+ifisize-1) = uaug*H_FIT
        END DO

        DO j = 1, FSIZE
            IF(FKOMB(i, j) == 1) THEN
                P1(iBeginRow:iEndRow, ivirs*ifisize+j) = MATMUL(BETA, (FSUPFINFKONF(j, :)))
            ELSE
                P1(iBeginRow:iEndRow, ivirs*ifisize+FSIZE+j) = MATMUL(BETA,
(FSUPFINFKONF(FSIZE+j, :)))
            END IF
        END DO
    END DO
END IF

END IF

END SUBROUTINE

```

C priedas. Publikacijų bendraautorių sutikimai teikti publikacijas disertacijos gynimui

PUBLIKACIJŲ BENDRAAUTORIAUS SUTIKIMAS

Aš, Juozas Atkočiūnas publikacijų

- Atkočiūnas, J.; Merkevičiūtė, D.; Venskųs, A.; Nagevičius, J. Prisisaikančiųjų santvarų op-timizavimo uždavinių matematiniai modeliai judamosios apkrovos atveju. Ūkio technologinis ir eko-nominis vystymas. 2(13): 93–99, 2007. ISSN 1392–8619.
- Atkočiūnas, J.; Merkevičiūtė, D.; Venskųs, A.; Skaržauskas, V. Nonlinear programming and optimal shakedown design of frames. Mechanika. 2(64): 27–33, 2007. ISSN 1392–1207.
- Atkočiūnas, J.; Merkevičiūtė, D.; Venskųs, A. Optimal shakedown design of bar systems: Strength, stiffness and stability constraints. Computers and Structures. 17–18(86): 1757–1768, 2008. ISSN 0045–7949.
- Atkočiūnas, J.; Venskųs, A. Optimal shakedown design of frames under stability conditions according to standards. Computers and Structures. 3–4(89): 435–443, 2011. ISSN 0045–7949.
- Kalanta, S.; Atkočiūnas, J.; Venskųs, A. Discrete optimization problems of the steel bar structures. Engineering Structures. 6(31): 1298–1304, 2009. ISSN 0141–0296.
- Venskųs, A.; Atkočiūnas, J.; Kalanta, S.; Ulitinai, T. Integrated load optimization of elastic-plastic axisymmetric plates at shakedown. Journal of Civil Engineering and Management. 2(16): 203–208, 2010. ISSN 1392–3730.
- Atkočiūnas, J.; Merkevičiūtė, D.; Venskųs, A.; Rimkus, L. Optimal shakedown design of steel structures. Anwendung der Optimierung in der nichtlinearen Tragwerksanalyse. Schriftenreihe des Institutes fuer Konstruktiven Ingenieurbau Bauhaus–Universität Weimar. Weimar : Bauhaus–Universität Weimar, Fakultät Baingenieurwesen, 137–144, 2007. ISBN 978–3–86068–307–1.
- Atkočiūnas, J.; Venskųs, A. Shakedown theory: a generalized approach to the optimization of elastic-plastic structures, in abstracts of the 4rd International conference of SDSMS'04: Strength, durability and stability of materials and structures. Kaunas: Technologija, 16–17, 2007. ISBN 978–9955–25–308–2
- Atkočiūnas, J.; Venskųs, A. Optimal shakedown design of frames under stability conditions, proceedings of the Ninth International Conference on Computational Structures Technology. Stirlingshire: Civil–Comp Press, 159, 2008. ISBN 978–1–905088–21–8.
- Atkočiūnas, J.; Venskųs, A. Optimal shakedown design of frames under stability conditions, Proceedings of the Ninth International Conference on Computational Structures Technology. Stirlingshire: Civil–Comp Press, 1–12, 2008. ISBN 978–1905–08–822–5.
- Kalanta, S.; Atkočiūnas, J.; Venskųs, A. Discrete optimization problems of the steel bar structures, selected papers of the 9th international conference “Modern building materials, structures and techniques”. Vilnius: Technika, 949–954, 2007. ISBN 978–9955–28–200–6.
- Kalanta, S.; Atkočiūnas, J.; Venskųs, A. Discrete optimization problems of the steel bar structures, in proceedings of the 9th international conference “Modern building materials, structures and techniques”, [CD]. Vilnius: Technika, 1–6, 2007. ISBN 978–9955–28–189–4.
- Kalanta, S.; Atkočiūnas, J.; Venskųs, A. Discrete optimization problems of the steel bar structures, abstracts of the 9th international conference Modern building materials, structures and techniques. Vilnius: Technika, 2007, 405–406, 2007. ISBN 978–9955–28–131–3.
- Venskųs, A.; Atkočiūnas, J. Plokštės analizės netiesinio uždavinio sprendimo programa "Rūta" ir jos integracijos galimybės, iš Statyba: 8–osios Lietuvos jaunųjų mokslininkų konferencijos "Lietuva be mokslo – Lietuva be ateities" medžiaga. Vilnius: Technika 277–282, 2005. ISBN 9986–05–893–7.
- Venskųs, A.; Atkočiūnas, J. Patobulintas prisitaikančių sistemų optimizacijos uždavinių sprendimo algoritmas, iš Statyba: 9–osios Lietuvos jaunųjų mokslininkų konferencijos "Mokslas – Lietuvos ateitis" medžiaga. Vilnius: Technika, 265–270, 2006. ISBN 9955–28–047–6.

bendraautorius, pripažįstu Artūro Venskaus autorinį indėlį bendrose publikacijose ir neprieštarauju, kad bendraautorius publikacijose skelbtą medžiagą jo autorinio indėlio ribose teiktų savo mokslo daktaro disertacijoje “Prisitaikančių konstrukcijų optimizavimas. Sąsajos su projektavimo standartais”.

Juozas Atkočiūnas



(parašas)

2011 m. kovo 28d.

PUBLIKACIJŲ BENDRAAUTORIAUS SUTIKIMAS

Aš, Stanislovas Kalanta publikacijų

- Kalanta, S.; Atkočiūnas, J.; Venskųs, A. Discrete optimization problems of the steel bar structures. Engineering Structures. 6(31): 1298–1304, 2009. ISSN 0141–0296.
- Venskųs, A.; Atkočiūnas, J.; Kalanta, S.; Ulitinas, T. Integrated load optimization of elastic-plastic axisymmetric plates at shakedown. Journal of Civil Engineering and Management. 2(16): 203–208, 2010. ISSN 1392-3730.
- Kalanta, S.; Atkočiūnas, J.; Venskųs, A. Discrete optimization problems of the steel bar structures, selected papers of the 9th international conference “Modern building materials, structures and techniques”. Vilnius: Technika, 949–954, 2007. ISBN 978–9955–28–200–6.
- Kalanta, S.; Atkočiūnas, J.; Venskųs, A. Discrete optimization problems of the steel bar structures, in proceedings of the 9th international conference “Modern building materials, structures and techniques”, [CD]. Vilnius: Technika, 1–6, 2007. ISBN 978–9955–28–189–4.
- Kalanta, S.; Atkočiūnas, J.; Venskųs, A. Discrete optimization problems of the steel bar structures, abstracts of the 9th international conference Modern building materials, structures and techniques. Vilnius: Technika, 2007, 405–406, 2007. ISBN 978–9955–28–131–3.

bendraautorius, pripažįstu Artūro Venskaus autorinį indėlį bendrose publikacijose ir neprieštaruju, kad bendraautorius publikacijose skelbtą medžiagą jo autorinio indėlio ribose teiktų savo mokslo daktaro disertacijoje “Prisitaikančių konstrukcijų optimizavimas. Sąsajos su projektavimo standartais”.

Stanislovas Kalanta

(parašas)

2011 m. kovo 28d.

PUBLIKACIJŲ BENDRAAUTORIAUS SUTIKIMAS

Aš, Dovilė Merkevičiūtė publikacijų

- Atkočiūnas, J.; Merkevičiūtė, D.; Venskų, A.; Nagevičius, J. Prisitaikančiųjų santvarų optimizavimo uždavinių matematiniai modeliai judamosios apkrovos atveju. Ūkio technologinis ir ekonominis vystymas. 2(13): 93–99, 2007. ISSN 1392–8619.
- Atkočiūnas, J.; Merkevičiūtė, D.; Venskų, A.; Skaržauskas, V. Nonlinear programming and optimal shakedown design of frames. Mechanika. 2(64): 27–33, 2007. ISSN 1392–1207.
- Atkočiūnas, J.; Merkevičiūtė, D.; Venskų, A. Optimal shakedown design of bar systems: Strength, stiffness and stability constraints. Computers and Structures. 17–18(86): 1757–1768, 2008. ISSN 0045–7949.
- Atkočiūnas, J.; Merkevičiūtė, D.; Venskų, A.; Rimkus, L. Optimal shakedown design of steel structures. Anwendung der Optimierung in der nichtlinearen Tragwerksanalyse. Schriftenreihe des Institutes fuer Konstruktiven Ingenieurbau Bauhaus–Universität Weimar. Weimar : Bauhaus–Universität Weimar, Fakultät Bauingenieurwesen, 137–144, 2007. ISBN 978–3–86068–307–1.

bendraautorius, pripažįstu Artūro Venskaus autorinį indėlį bendrose publikacijose ir neprieštarauju, kad bendraautorius publikacijose skelbtą medžiagą jo autorinio indėlio ribose teiktų savo mokslo daktaro disertacijoje “Prisitaikančių konstrukcijų optimizavimas. Sąsajos su projektavimo standartais”.

Dovilė Merkevičiūtė



(parašas)

2011 m. kovo 28d.

PUBLIKACIJOS BENDRAAUTORIAUS SUTIKIMAS

Aš, Juozas Nagevičius publikacijos

- Atkočiūnas, J.; Merkevičiūtė, D.; Venskų, A.; Nagevičius, J. Prisitaikančiųjų santvarų optimizavimo uždavinių matematiniai modeliai judamosios apkrovos atveju. Ūkio technologinis ir ekonominis vystymas. 2(13): 93–99, 2007. ISSN 1392–8619.

bendraautorius, pripažįstu Artūro Venskaus autorinį indėlį bendroje publikacijoje ir neprieštarauju, kad bendraautorius publikacijoje skelbtą medžiagą jo autorinio indėlio ribose teiktų savo mokslo daktaro disertacijoje “Prisitaikančių konstrukcijų optimizavimas. Sąsajos su projektavimo standartais”.

Juozas Nagevičius


(parašas)

2011 m. kovo 28d.

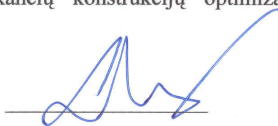
PUBLIKACIJOS BENDRAAUTORIAUS SUTIKIMAS

Aš, Liudvikas Rimkus publikacijos

- Atkočiūnas, J.; Merkevičiūtė, D.; Venskų, A.; Rimkus, L. Optimal shakedown design of steel structures. Anwendung der Optimierung in der nichtlinearen Tragwerksanalyse. Schriftenreihe des Institutes fuer Konstruktiven Ingenieurbau Bauhaus-Universität Weimar. Weimar : Bauhaus-Universität Weimar, Fakultät Bauingenieurwesen, 137–144, 2007. ISBN 978–3–86068–307–1.

bendraautorius, pripažįstu Artūro Venskaus autorinį indėlį bendroje publikacijoje ir neprieštaruju, kad bendraautorius publikacijoje skelbtą medžiagą jo autorinio indėlio ribose teiktų savo mokslo daktaro disertacijoje “Prisitaikančių konstrukcijų optimizavimas. Sąsajos su projektavimo standartais”.

Liudvikas Rimkus



(parašas)

2011 m. kovo 28d.

PUBLIKACIJOS BENDRAAUTORIAUS SUTIKIMAS

Aš, Valentinas Skaržauskas publikacijos

- Atkočiūnas, J.; Merkevičiūtė, D.; Venskų, A.; Skaržauskas, V. Nonlinear programming and optimal shakedown design of frames. *Mechanika*. 2(64): 27–33, 2007. ISSN 1392–1207.

bendraautorius, pripažįstu Artūro Venskaus autorinį indėlį bendroje publikacijoje ir neprieštarauju, kad bendraautorius publikacijoje skelbtą medžiagą jo autorinio indėlio ribose teiktų savo mokslo daktaro disertacijoje “Prisitaikančių konstrukcijų optimizavimas. Sąsajos su projektavimo standartais”.

Valentinas Skaržauskas



(parašas)

2011 m. kovo 28d.

PUBLIKACIJOS BENDRAAUTORIAUS SUTIKIMAS

Aš, Tomas Ulitinas publikacijos

- Venskųs, A.; Atkočiūnas, J.; Kalanta, S.; Ulitinas, T. Integrated load optimization of elastic-plastic axisymmetric plates at shakedown. Journal of Civil Engineering and Management. 2(16): 203–208, 2010. ISSN 1392-3730.

bendraautorius, pripažįstu Artūro Venskaus autorinį indėlį bendroje publikacijoje ir neprieštarauju, kad bendraautorius publikacijoje skelbtą medžiagą jo autorinio indėlio ribose teiktų savo mokslo daktaro disertacijoje “Prisitaikančių konstrukcijų optimizavimas. Sąsajos su projektavimo standartais”.

Tomas Ulitinas



(parašas)

2011 m. kovo 28d.