

A priedas. Rungės ir Kuto metodas

Rungės-Kutos metodu sprendiamos diferencialinės lygtys. Norint skaitiniu būdu išspręsti diferencialinę lygtį, reikia žinoti ieškomos funkcijos ir jos išvestinės reikšmės. Šios reikšmės, nusakytos viename taške, vadinamos pradinėmis sąlygomis, o pati lygtis su pradinėmis sąlygomis – Košy uždavinys. Dinaminio modelio judėjimo lygtys sprendžiamos ketvirtos eilės Rungės ir Kutos metodu. Šio metodo paklaida h^4 , kai h – mūsų ieškomo kintamojo žingsnis.

Nagrinėjamas Koši uždavinys. Sprendžiama diferencialinė lygtis, kuri bendru atveju užrašoma:

$$\left\{ \frac{dx}{dt} \right\} = \{F(t, x_1, x_2, x_3, x_4)\}; \quad (\text{A.1.})$$

kai x_k – būdenos parametrų vektoriaus kintamieji; $\{x(t = t_0)\} = \{x_0\}$;

Kintamojo t žingsnis h laikomas pastoviu ir žymimas:

$$t_k = t_0 + kh; \text{ kai } x_k(t_k) = x_k; \quad (\text{A.2.})$$

Ketvirtos eilės Rungės ir Kutos metodo būsenos parametro vektoriaus $\{X\}$ reikšmės užrašomos taip:

$$x_{k+1/2}^{(1)} = x_k + \frac{h}{2} f(t_k, x_k); \quad (\text{A.3.})$$

$$x_{k+1/2}^{(2)} = x_k + \frac{h}{2} f\left(t_k + \frac{h}{2}, x_{k+1/2}^{(1)}\right); \quad (\text{A.4.})$$

$$x_{k+1/2}^{(3)} = x_k + hf\left(t_k + \frac{h}{2}, x_{k+1/2}^{(2)}\right); \quad (\text{A.5.})$$

$$x_{k+1} = x_k + \frac{h}{6} (f(t_k, x_k) + 2f\left(t_k + \frac{h}{2}, x_{k+1/2}^{(1)}\right) + 2f\left(t_k + \frac{h}{2}, x_{k+1/2}^{(2)}\right) + f(t_{k+1}, x_{k+1}^{(1)})); \quad (\text{A.6.})$$

Dinaminių modelių mechaninės sistemos yra daugiamatės, kadangi šie dinaminiai modeliai turi kelis įėjimo ir išėjimo parametrus, todėl jas galima užrašyti vektorine forma. Būsenos vektorinė diferencialinė lygtis sužrašoma tokia forma:

$$\{\dot{X}\} = [A]\{X\} + [B]\{u\}; \quad (\text{A.7.})$$

čia $[A]$ ir $[B]$ matricos, $\{u\}^T$ – įėjimo parametrų vektorius.

Rungės-Kutos metodo algoritmas Fortran programa

1. laiko žingsnio perskaičiavimas

```
A(1)=0.5*H
A(2)=A(1)
A(3)=H
A(4)=H
A(5)=A(1)
TE=T
```

2. reikšmių priskyrimas

```
DO J=1,N
  Y(J)=X(J)
  YF(J)=X(J)
END DO
```

3. naujos reikšmės paskaičiavimas

```
DO I=1,4
  CALL LSK(TE,Y,YR,N)
  TE=T+A(I)
  DO K=1,N
    Y(K)=X(K)+A(I)*YR(K)
    YF(K)=YF(K)+A(I+1)*YR(K)/3.0D0
  END DO
END DO
```

4. naujos reikšmės priskyrimas

```
DO J=1,N
  X(J)=YF(J)
END DO
```

B priedas. Niutono metodas

Sprendžiama lygčių sistema:

$$[H] \cdot \Delta\{\varphi\} = -\{J\}, \quad (\text{B.1.})$$

čia

$$[H] = \begin{bmatrix} \frac{\partial^2 \Phi}{\partial \varphi_1 \partial \varphi_1} & \dots & \frac{\partial^2 \Phi}{\partial \varphi_1 \partial \varphi_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 \Phi}{\partial \varphi_m \partial \varphi_n} & \dots & \frac{\partial^2 \Phi}{\partial \varphi_m \partial \varphi_n} \end{bmatrix} - \text{Heses matrica}; \quad (\text{B.2.})$$

$$\{J\} = \left\{ \frac{\partial \Phi}{\partial \varphi_1} \dots \frac{\partial \Phi}{\partial \varphi_n} \right\} - \text{Jakobio matrica stulpelis}; \quad (\text{B.3.})$$

$\Delta\{\varphi\}$ - nežinomųjų vektorius pokytis.

Sistemos sprendinys:

$$\Delta\{\varphi\} = [H]^{-1} \{J\} \quad (\text{B.4.})$$

Niutono metodo algoritmas užrašytas Maple programa:

```
z:=10:
while Z>0.01 do
    if (abs(Phi)>0.01) then
        H1:=MatrixInverse(H);
        d_phi:=MatrixVectorMultiply(H1,J);
        phi1:= phi+d_phi;
        Z:=
max(seq(abs(evalf(d_phi[i])/evalf(phi1[i])),i=1..m)):
        for i to m do phi[i]:=phi1[i]: end do;
    else
        Z:=Phi;
    end if;
    print("Z=",Z, "Phi=", Phi);
print(seq(phi[i],i=1..m));
end do;
```

C priedas. Sukurto modelio kodas FORTRAN programine kalba

```

!*****
! tiesi atkarpa su atsišakojimais ir valdymu
!*****
program KPMSV_2
implicit real*8 (a-h,o-z)
PARAMETER KSK=5,KEL=1000

Integer npoint

Dimension dv(kel), dk(kel), X(kel)
DIMENSION gkelem(KEL,14),ckv(4),ask(kel)
DIMENSION
elemL(KEL),gknr(KEL),gkilgis(KEL),nelem(KEL),MazgasNr(KEL)
DIMENSION ck(KEL),ckttau(KEL),v(KEL),vttau(KEL)
Dimension F_CO_0(ksk,kel), F_CH_0(ksk,kel), F_Nox_0(ksk,kel),
F_Pm_0(ksk,kel), F_CO2_0(ksk,kel)

DIMENSION SAIN(KEL,3),SAOUT(KEL,3)
DIMENSION IEKS(KEL),ISKS(KEL)
INTEGER SS,SN,IEKS,ISKS,SUM1,SUM2 !ISKN, IEKN,

CHARACTER *80 TEXT *80

external TSL2

common /eismo_juostos_parametrai_1/
vmax(1000),ckmax(1000),qmax(1000),p_in(1000)
common /eismo_juostos_parametrai_2/
p_out(1000),rk_in(1000),rk_out(1000),rv_in(1000)
common /eismo_juostos_parametrai_3/
rv_out(1000),v_min(1000),ck_min(1000),q_min(1000)
COMMON /keliu_struktura/ dL(1000),gkelstr(1000,7)

COMMON /lygtims/ dt,kkeliu,AKKM(1000,6),npoint
common /parametrai/
tau_max(500),vtau(1000,2000),cktau(1000,2000)
COMMON /Laikas/ t

!call AKEA_3(KEL,KKELIU,ELEMSK,NPOINT,GKELEM,AKKM)
call AKEA_3(ELEMSK,NPOINT,GKELEM)
call sankryza1(KEL,SAIN,SAOUT,SUM1,SUM2,ss)

write(6,*) ' '
write(6,*) ' '
PAUSE 'GALIMA PAKEISTI DUOMENIS KELIU PARAMETRU LENTELEJE'

```

```

!***galima pakoreaguoti duomenis
FILE='Keliu_parametru_lentele_2.DAT'*****

OPEN(UNIT=2,FILE='Keliu_parametru_lentele_2.DAT')
!*****
*****

!***pradiniai duomenys ir kraštinės sąlygos
skaičiavimams*****

OPEN(UNIT=8,FILE='pradiniai_duomenys.DAT')
OPEN(UNIT=9,FILE='krastines_salygos.dat')
!*****
*****

!***skaičiavimų rezultatu
failai*****

OPEN(UNIT=10,FILE='koncentracija_rez1.DAT')
OPEN(UNIT=11,FILE='greitis_rez1.DAT')
OPEN(UNIT=12,FILE='koncentracija_rez2.DAT')
OPEN(UNIT=13,FILE='greitis_rez2.DAT')
!*****
*****

READ(2,*) TEXT
write(6,*) ' '
WRITE(6,'(a)') TEXT

do j=1,NPOINT

    READ(2,*)
    gknr(j),MazgasNr(j),vmax(j),ckmax(j),qmax(j),v_min(j),ck_min(j),q_
min(j),p_in(j),p_out(j),rk_in(j),rk_out(j),rv_in(j),rv_out(j)
        WRITE(6,1002)
    gknr(j),MazgasNr(j),vmax(j),ckmax(j),qmax(j),v_min(j),ck_min(j),q_
min(j),p_in(j),p_out(j),rk_in(j),rk_out(j),rv_in(j),rv_out(j)

end do
CLOSE (2,STATUS='KEEP')
write(6,*) ' '
pause 'Nuskaityti KELIU PARAMETRU LENTELES duomenys'

READ(8,*) TEXT
write(6,*) ' '
WRITE(6,'(a)') TEXT

    READ(8,*)          ckprad, vprad, t0,    tmax,    dt, nprint
    WRITE(6,1002)      ckprad, vprad, t0,    tmax,    dt, nprint

```

```

CLOSE (8,STATUS = 'KEEP')

      READ(9,*)          (ckv(i),i=1,4) !v_iej,   v_isej, ck_iej, ck_isej
      WRITE(6,1002)      (ckv(i),i=1,4) !v_iej,   v_isej, ck_iej, ck_isej

CLOSE (9,STATUS = 'KEEP')
1002    FORMAT(14(G11.5))
write(6,*) ' '
pause 'pradiniai duonenys ir krastines salygos'

do i=1,npoint
    ck(i)=ckprad
    v(i)=vprad
end do

do i=1,kkeliu
    do j=AKKM(i,2),AKKM(i,3)
        tau_max(j)=DNINT(AKKM(i,6)/v_min(j))
    end do

    do il=AKKM(i,2),AKKM(i,3)
        do j1=1,tau_max(il)
            vtau(il,j1)=0.0d0 !v_min(il)
            cktau(il,j1)=0.0d0 !ck_min(il)
        end do
    end do

    do il=AKKM(i,2),AKKM(i,3)
        F_CO_0(i,il) =0.0d0
        F_CH_0(i,il) =0.0d0
        F_NoX_0(i,il)=0.0d0
        F_Pm_0(i,il) =0.0d0
        F_CO2_0(i,il)=0.0d0
    end do
end do

!-----
isk=0
h1=0.0d0

do i=1,npoint
    X(2*i-1)=ck(i)
    X(2*i)=v(i)
end do

do t=t0,tmax,dt

    H1=H1+dt

    call valdymas1(kel,X,ckv,ss,h1,SAIN,SAOUT)

```

```

        nk=npoint*2

        call RUNKUT(nk,dt,T,X,TSL2)

do i1=1,kkeliu
    do i=AKKM(i1,2),AKKM(i1,3)
        if (X(2*i-1).Le.ck_min(i)) then
            X(2*i-1)=ck_min(i)
        else if (X(2*i-1).ge.ckmax(i)) then
            X(2*i-1)=ckmax(i)
        else
            X(2*i-1)=X(2*i-1)
        end if

        if (X(2*i).Le.v_min(i)) then
            X(2*i)=v_min(i)
        else if (X(2*i).ge.vmax(i)) then
            X(2*i)=vmax(i)
        else
            X(2*i)=X(2*i)
        end if
        ! atitinkamai ck(i),v(i)
    end do

    call AKEL(isk,nprint,t,X,ask)

    call var_emisija2(ksk,kel,isk,nprint,t,X,ask,
F_CO_0,F_CH_0,F_Nox_0,F_Pm_0,F_CO2_0)

    if (isk.eq.nprint) then

        write(6,*) ' t= ', t
        write(6,*) ' '

        do i=AKKM(i1,2),AKKM(i1,3)

write(6,*) X(2*i-1),X(2*i) !atitinkamai ck(i),v(i)
        end do
        write(6,*) ' '

write(10,1003) t,(X(2*i-1),i=AKKM(1,2),AKKM(1,3))
write(11,1003) t,(X(2*i),i=AKKM(1,2),AKKM(1,3))

        isk=0
    else
        isk=isk+1
    end if

end do
end do

```

```
1003 FORMAT (1X, 200(G12.5))
```

```
Stop
end program KPMSV_2
```

```
! *****
!      PAPROGREMĖ KELIO ELEMENTAMS SUKURTI IR INFORMACIJAI APRAŠYTI
!
! *****
!
!      KELIO ELEMENTŲ, MAZGŲ, RIBINIŲ PARAMETRŲ LENTELĖ          -
GKELEM
!      ELEMENTO ILGIS                      - dL
!      KELIO NUMERIS                      - GKNR
!      KELIO ILGIS                        -
GKILGIS
!      ELEMENTŲ SKAIČIUS                  -
NELEM
!      MAKSIMALUS GALIMAS GREITIS MAZGE    - VMAX
!      MAKSIMALI KONCENTRACIJA MAZGE      -
CKMAX
!      KELIO KRAŠTINIAI MAZGAI             - AKKM
!      KELIŲ SKAIČIUS                     -
KKELIU
!      MAZGŲ NUMERIAI                     -
MAZG_NR
!
! *****
```

```
      subroutine AKEA_3(ELEMSK,MAZG_NR,GKELEM)
      implicit real*8 (a-h,o-z)

      parameter kel=1000

      DIMENSION gkelem(KEL,14) !,gkelstr(KEL,7) !,aKKM(KEL,5)
      DIMENSION gknr(KEL),gkilgis(KEL),nelem(KEL),mazgSk(KEL)
      DIMENSION vmax(KEL),ckmax(KEL),qmax(KEL)
      DIMENSION p_in(KEL),p_out(KEL),rk_in(KEL),rk_out(KEL)
      DIMENSION rv_in(KEL),rv_out(KEL),dL1(kel),dL0(kel)

      CHARACTER *80 TEXT *80

      COMMON /keliu_struktura/
      dL(1000),gkelstr(1000,7)
      COMMON /lygtims/
      dt,kkeliu,AKKM(1000,7),npoin

      OPEN(UNIT=1,FILE='Kelias_2.DAT')
      OPEN(UNIT=2,FILE='Keliu_parametru_lentele_2.DAT')
      OPEN(UNIT=3,FILE='KELIU_KRASTINIAI_MAZGAI.DAT')
```

```

OPEN (UNIT=4, FILE='KELIU_STRUKTURA.DAT')

      READ(1,*) TEXT
      WRITE(6,'(a)') TEXT

      READ(1,*) kkeliu
      write(6,1000) kkeliu

      READ(1,*) TEXT
      WRITE(6,'(a)') TEXT

      DO i=1, kkeliu
        READ(1,*)
        gknr(i), gKilgis(i), nelem(i), MazgSk(i), vmax(i), ckmax(i), qmax(i), p_i
n(i), p_out(i), rk_in(i), rk_out(i), rv_in(i), rv_out(i)
          WRITE(6,1000)
        gknr(i), gkilgis(i), MazgSk(i), vmax(i), ckmax(i), qmax(i), p_in(i), p_ou
t(i), rk_in(i), rk_out(i), rv_in(i), rv_out(i)

      END DO

      WRITE(4,1003)'      '
      WRITE(2,1001)'      '

      ELEMSK=0
      MAZG_NR=0
      ELEMSK1=0
      MAZG_NR1=0

      do j=1, kkeliu

        dL1(j)=gkilgis(j)/nelem(j)

        do k=1, nelem(j)+1

          dL0(k)=dL1(j)

        end do

        MN=2
        MN1=1

      DO il=1, nelem(j)
        ELEMSK1=elemsk1+MN1+1
        MAZG_NR1=MAZG_NR1+MN1+1

        gkelstr(il,1)=gknr(j)
        gkelstr(il,2)=ELEMSK1-1      !elemento k-i numeris
        gkelstr(il,3)=ELEMSK1      !elemento i-j numeris
        gkelstr(il,4)=MAZG_NR1-2    !mazgas k
        gkelstr(il,5)=MAZG_NR1-1    !mazgas i

```

```

gkelstr(i1,6)=MAZG_NR1      !mazgas j
gkelstr(i1,7)=dL0(j)        !elemento ilgis

MN=0
MN1=0

WRITE(4,1002) (gkelstr(I1,J1),J1=1,7)

END DO

DO i=1,mazgSk(j)

  ELEMSK=ELEMSK+1
  MAZG_NR=MAZG_NR+1

  gkelem(i,1)=gknr(j)
  gkelem(i,2)=MAZG_NR      !mazgas i
  gkelem(i,3)=vmax(j)
  gkelem(i,4)=ckmax(j)
  gkelem(i,5)=qmax(j)
  gkelem(i,6)=0.1          !vmin    mazge i
  gkelem(i,7)=0.0          !ckmin   mazge i
  gkelem(i,8)=0.0          !qmin    mazge i
  gkelem(i,9)=p_in(j)
  gkelem(i,10)=p_out(j)
  gkelem(i,11)=rk_in(j)
  gkelem(i,12)=rk_out(j)
  gkelem(i,13)=rv_in(j)
  gkelem(i,14)=rv_out(j)

  WRITE(2,1002) (gkelem(I,J1),J1=1,14)

  IF(I.eq.1) THEN
    AKKM(J,1)=J
    aKKM(J,2)=gkelem(i,2)
  ELSE IF (I.eq.mazgSk(j)) THEN
    aKKM(J,3)=gkelem(i,2)
  END IF
  aKKM(J,4)=NELEM(J)
  aKKM(J,5)=NELEM(J)+1
  aKKM(J,6)=dL0(j)

END DO

end do

WRITE(3,2000) ((aKKM(I,J),J=1,6),I=1,kkeliu)
WRITE(6,2000) ((aKKM(I,J),J=1,6),I=1,kkeliu); pause "-1-"
CLOSE (1,STATUS='KEEP')
CLOSE (2,STATUS='KEEP')
CLOSE (3,STATUS='KEEP')

```

```

CLOSE (4, STATUS = 'KEEP')

200      FORMAT(1X,8(G10.0))
1000     FORMAT(1X,12(G11.5))
1001     FORMAT(1X,'Kelio_Nr    Mazgas_i    v_max(i)    k_max(i)
q_max(i)    v_min(i)    k_min(i)    q_min(i)    p_in(i)    p_out(i)
rk_in(i)    rk_out(i)    rv_in(i)    rv_out(i)'/G11.30)
1002     FORMAT(14(G11.5))
1003     FORMAT('Kelio_Nr    El(k-i)Nr    El(i-j)Nr    Mazgas_k
Mazgas_i    Mazgas_j    Elem_ilg'/G11.30)
2000     FORMAT(1X,'Kelio_Nr    Mazgas_IN    Mazgas_OUT    ELEM_SK
MAZG_SK    kel_elem_ilg'/'      6(G11.5))

RETURN
END subroutine AKEA_3

subroutine TSL2(t,Y1,YR1,taskai)
implicit real*8 (a-h,o-z)
Dimension ck(npoin), v(npoin), ckttau(npoin),
vttau(npoin),eps(npoin), f(npoin)
Dimension ck_in(npoin), ck_out(npoin), v_in(npoin),
v_out(npoin), dv(npoin), dk(npoin)
Dimension gknr(npoin),Y1(2*npoin),YR1(2*npoin),dL(npoin)

integer taskai, task

common /eismo_juostos_parametrai_1/
vmax(1000),ckmax(1000),qmax(1000),p_in(1000)
common /eismo_juostos_parametrai_2/
p_out(1000),rk_in(1000),rk_out(1000),rv_in(1000)
common /eismo_juostos_parametrai_3/
rv_out(1000),v_min(1000),ck_min(1000),q_min(1000)

COMMON /lygtims/          dt, kkeliu, AKKM(1000,6), npoin

do i1=1, kkeliu

    do i=AKKM(i1,2), AKKM(i1,3)
        ck(i)=Y1(2*i-1)
        v(i) =Y1(2*i)
        dL(i)=AKKM(i1,6)
    end do

end do

call velinimas2(v,ck,vttau,ckttau,dL)

do i1=1, kkeliu

    m1=6

```

```

m2=10
gama2=2.5d0
gama3=5.5d0
do j=1,npoint
    eps(j)=ck(j)/ckmax(j)
end do

do i=AKKM(i1,2),AKKM(i1,3)

    if (i.eq.AKKM(i1,2)) then !--- pirmame mazge

        ck_in(i)=0.0d0
        ck_out(i)=p_out(i)*rk_out(i)*(1.0d0-
ck(i+1)/ckmax(i+1))*(ck(i)*v(i)/qmax(i))*ck(i)

        dk(i)=ck_in(i)-ck_out(i)

        v_in(i)=0.0d0
        v_out(i)=p_out(i)*rv_out(i)*(v(i)+v(i+1))/(2*dL(i))*(1.0d0-
ck(i+1)/ckmax(i+1))*m1*v(i)-
v(i)/vmax(i)*dexp(gama3*(ck(i)/ckmax(i))*m2*v(i)/vmax(i))

        dv(i)=v_in(i)+v_out(i)

    else if (i.eq.AKKM(i1,3)) then !--- paskutiniame mazge

        ck_in(i)=p_in(i)*rk_in(i)*(1.0d0-
ck(i)/ckmax(i))*(ckttau(i-1)*vttau(i-1))/qmax(i-1)*ck(i)
        ck_out(i)=0.0d0

        dk(i)=ck_in(i)-ck_out(i)

        v_in(i)=p_in(i)*rv_in(i)*vttau(i-1)/dL(i-1)*(1.0d0-
ck(i)/ckmax(i))*v(i)
        v_out(i)=0.0d0
        dv(i)=v_in(i)+v_out(i)

    else !---- tarpiniuose mazguose

        ck_in(i)=p_in(i)*rk_in(i)*(1.0d0-
ck(i)/ckmax(i))*(ckttau(i-1)*vttau(i-1))/qmax(i-1)*ck(i)
        ck_out(i)=p_out(i)*rk_out(i)*(1.0d0-
ck(i+1)/ckmax(i+1))*(ck(i)*v(i)/qmax(i))*ck(i)

        dk(i)=ck_in(i)-ck_out(i)

        v_in(i)=p_in(i)*rv_in(i)*vttau(i-1)/dL(i-1)*(1.0d0-
ck(i)/ckmax(i))*v(i)

        v_out(i)=p_out(i)*rv_out(i)*(v(i)+v(i+1))/(2.0d0*dL(i))*(1.0d0-

```

```

ck(i+1)/ckmax(i+1))**m1*v(i)-
v(i)/vmax(i)*dexp(gama3*(ck(i)/ckmax(i))**m2*v(i)/vmax(i))

      if (eps(i).LT.0.0d0.and.eps(i).LT.eps(i+1)) then

          f(i)=dexp(gama2*(1.0d0-
eps(i+1)/eps(i))*eps(i)*sign(1.0d0,p_out(i))*sign(1.0d0,eps(i+1)/e
ps(i)))

      else

          f(i)=0.0d0

      end if

      dv(i)=v_in(i)+f(i)+v_out(i)

end if

end do

do i=AKKM(i1,2),AKKM(i1,3)
    YR1(2*i-1)=dk(i)
    YR1(2*i) =dv(i)
end do
end do

return
end subroutine TSL2

subroutine velinimas2(v,ck,vttau,ckttau,dL)
implicit real*8 (a-h,o-z)
DIMENSION vttau(npoin),ckttau(npoin)
DIMENSION V(npoin),ck(npoin)DIMENSION tau(npoin),dL(npoin)

common /eismo_juostos_parametrai_3/
rv_out(1000),v_min(1000),ck_min(1000),q_min(1000)

COMMON /lygtims/ dt,kkeliu,AKKM(1000,6),npoin
common /parametrai/
tau_max(500),vtau(1000,2000),cktau(1000,2000)

do i=1,kkeliu

    do j=AKKM(i,2),AKKM(i,3)

        tau(j)=DNINT(AKKM(j,6)/v(j))
        if (tau(j).ge.tau_max(j)) then
            tau(j)=tau_max(j)-1.0d0
        end if
    
```

```

        do j1=2,tau_max(j)
            vtau(j,j1-1)=vtau(j,j1)
            cktau(j,j1-1)=cktau(j,j1)
        end do
        vtau(j,tau_max(j))=v(j)
        cktau(j,tau_max(j))=ck(j)
    end do

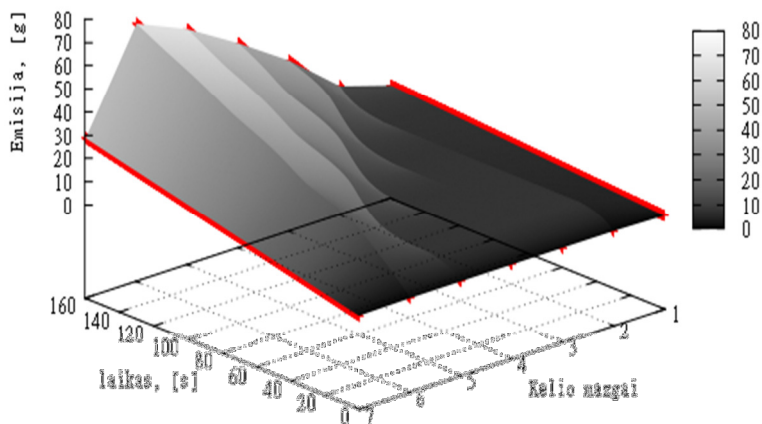
    do j=AKKM(i,2),AKKM(i,3)
        k=tau_max(j)
        k1=tau(j)
        if (dt.GE.tau(j)) then
            vttau(j)=vtau(j,k)
            ckttau(j)=cktau(j,k)
        else
            vttau(j)=vtau(j,k-k1)
            ckttau(j)=cktau(j,k-k1)
        end if
    end do

end do
100 FORMAT (1X, 'A Matrica' / 100(F7.2) /' ')
101 FORMAT (1X, 'Tau' / 100(F7.2) /' ')
102 FORMAT (1X, 't' , f7.2 /' ')
103 FORMAT (1X, 100(g12.5))

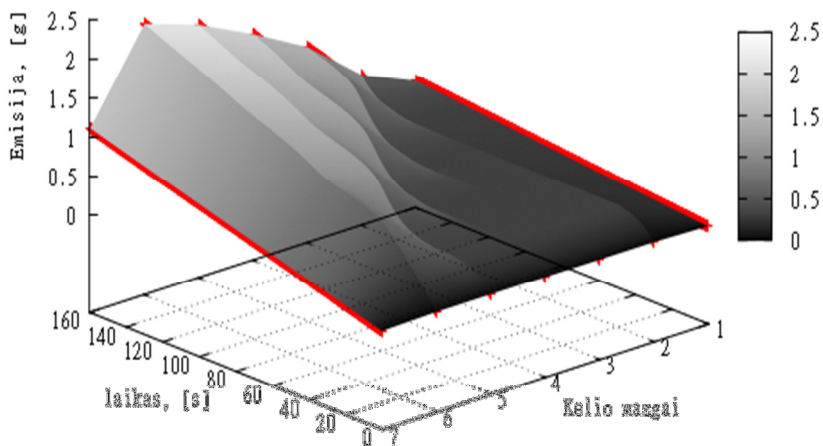
return
end subroutine Velinimas2

```

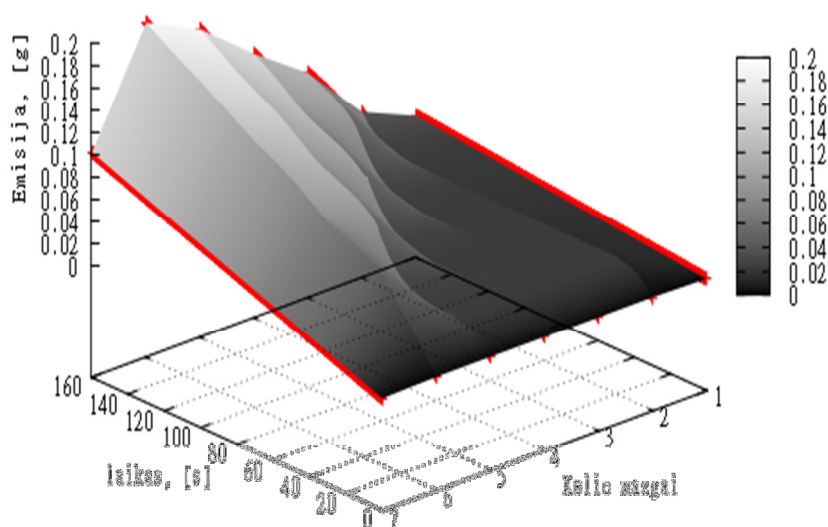
D priedas. Vidaus degimo variklio emisijų prognozės rezultatai



1D pav. VDV CO₂ emisijos kelio mazguose priklausomybė nuo laiko
Fig. 1D. Dependence of IC engine CO₂ emissions on time at road points

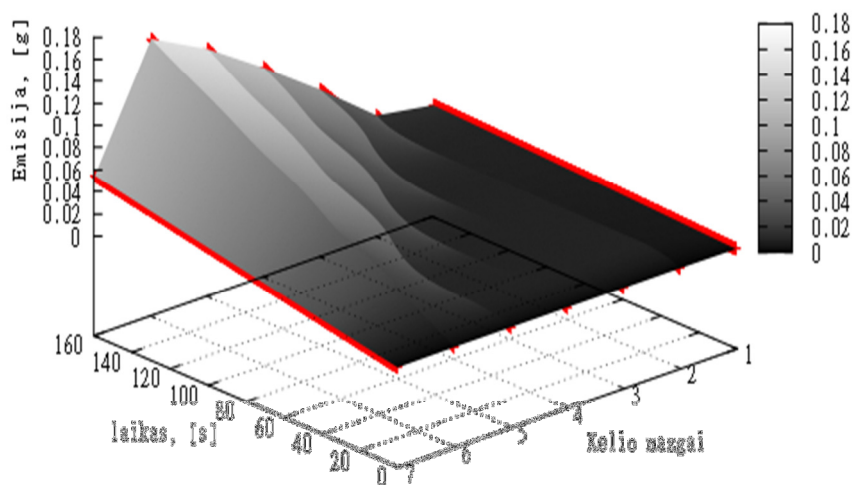


2D pav. VDV CO emisijos kelio mazguose priklausomybė nuo laiko
Fig. 2D. Dependence of IC engine CO emission on time at road points



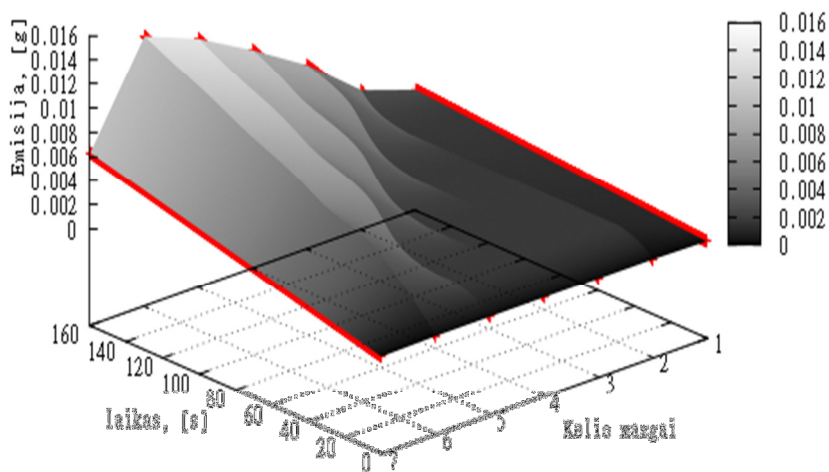
3D pav. VDV CH emisijos kelio mazguose priklausomybė nuo laiko

Fig. 3D. Dependence of IC engine CH emission on time at road points

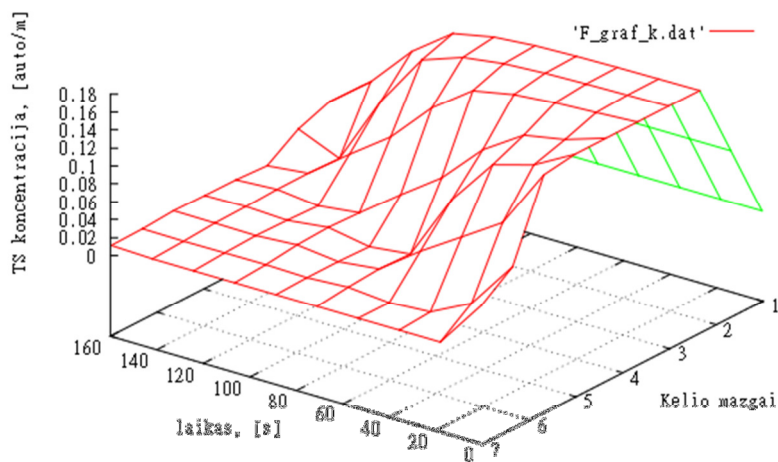


4D pav. VDV NOx emisijos kelio mazguose priklausomybė nuo laiko

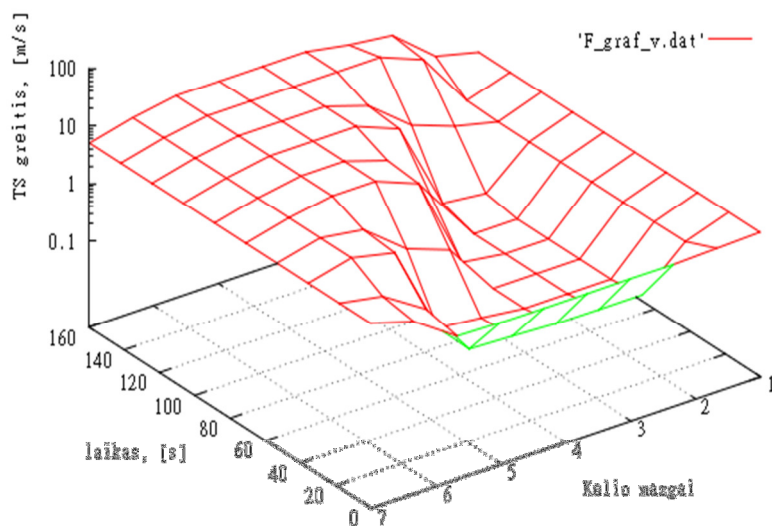
Fig. 4D. Dependence of IC engine NOx emission on time at road points



5D pav. Vidaus degimo variklio Pm emisijos kelio mazguose priklausomybė nuo laiko
Fig. 5D. Dependence of internal combustion engine Pm emission on time at road points



6D pav. Automobilių srauto koncentracijos kelio mazguose priklausomybė nuo laiko
Fig. 6D. Dependence of vehicles' flow concentration on time at road points



7D pav. Automobilių srauto greičio kelio mazguose priklausomybė nuo laiko
Fig. 7D. Dependence of vehicles' flow speed on time at road points