```
> restart:
  with(plots):
  with(Statistics):
  #

  #-----------------------------------------------------------------
  ----
  #-------------------------   ---------    -----------
  ----
  #-----------------------------------------------------------------
  ----
  solverStupen:=proc(stepPoros, coefPoros, Perm, Xdiscont, pl0,
  pL0, SetkaX, SetkaT)
            #stepPoros  -      ,                 coefPoros -

            #Perm  -
            #Xdiscont ,  p l 0 ,  p L 0  -     ,

            local i, j, N:
            local k, l, L, U, eq, rr, PstepNaOtrezke, Pstep, AA,
  GradPstep,DtPstep,StupenPl, alpha, beta, c, Kosh:
            local a, d:
            local NSetkaX, NSetkaT, shagX, shagT, UU, Ux, Ut;

            d:=stepPoros:
            a:=0:
            N:=numelems(Perm):

            #
            k:=MassiveToPiece(Perm, Xdiscont):

            l:=Xdiscont[1]:
            L:=Xdiscont[N+1]:

            #
            for i from 1 to N do
                      alpha[i]:=evalf(1/(Perm[i]*(d+1)*(d-a+2)*
  coefPoros)):
            end do:
            beta:=d+2:
            # ,
            c[0]:=coeff(pl0,t):


            #

            #
            eq[1]:=pl0-t*coeff(pl0,t)=c[0]*alpha[1]*l^(beta-a)+c[1,
  1]*l^(1-a)+c[2,1]:
            eq[2]:=pL0-t*coeff(pL0,t)=c[0]*alpha[N]*L^(beta-a)+c[1,
  N]*L^(1-a)+c[2,N]:

            #
            for i from 1 to N-1 do
                      eq[i+2]:=c[0]*alpha[i]*Xdiscont[i+1]^(beta-a)
  +c[1,i]*Xdiscont[i+1]^(1-a)+c[2,i]=
```

```
                                    c[0]*alpha[i+1]*Xdiscont[i+1]^
(beta-a)+c[1,i+1]*Xdiscont[i+1]^(1-a)+c[2,i+1];
          end do:

          #
          for j from 1 to N-1 do
                    eq[j+N+1]:=eval(k, x = Xdiscont[j+1]-10^(-4))
*(c[0]*alpha[j]*(beta-a)*Xdiscont[j+1]^(beta-a-1)+c[1,j]*(1-a)*
Xdiscont[j+1]^(-a))=
                    eval(k, x = Xdiscont[j+1]+10^(-4))*(c[0]*
alpha[j+1]*(beta-a)*Xdiscont[j+1]^(beta-a-1)+c[1,j+1]*(1-a)*
Xdiscont[j+1]^(-a));
          end do:

          #         k
          U:=c[0]*(ALPHA*x^(beta-a)+t)+C1*x^(1-a)+C2:

          #
          rr:=solve({seq(eq[i], i=1..2*N)}, {seq(c[1,i], i=1..N),
seq(c[2,i], i=1..N)}):

          #     ,         t

          PstepNaOtrezke:=seq(subs(subs(rr[i], C1=c[1, i]), subs
(rr[N+i], C2=c[2, i]), ALPHA=alpha[i], U), i=1..N):

          Matrix([[seq(x>=Xdiscont[i] and x<=Xdiscont[i+1], i = 1
.. N-1), x>=Xdiscont[N] and x<=Xdiscont[N+1]+1], [seq
(PstepNaOtrezke[i], i=1..N)]]):
          Pstep:=piecewise(seq(i, i in %)): print(%);
          Matrix([[seq(x>=Xdiscont[i] and x<=Xdiscont[i+1], i = 1
.. N-1), x>=Xdiscont[N] and x<=Xdiscont[N+1]+1], [seq(Perm[i]*
diff(PstepNaOtrezke[i], x), i=1..N)]]):
          GradPstep:=piecewise(seq(i, i in %)):
          Matrix([[seq(x>=Xdiscont[i] and x<=Xdiscont[i+1], i = 1
.. N-1), x>=Xdiscont[N] and x<=Xdiscont[N+1]+1], [seq(diff
(PstepNaOtrezke[i], t), i=1..N)]]):
          DtPstep:=piecewise(seq(i, i in %)):

          #
          evalf[5](Matrix([[seq(x>=Xdiscont[i] and x<=Xdiscont
[i+1], i = 1 .. N)], [seq(k[i], i=1..N)]])):
          StupenPl:=piecewise(seq(i, i in %)):

          #     (         )
          Kosh:=eval(Pstep,t=0):

          #
          NSetkaX:=numelems(SetkaX):
          NSetkaT:=numelems(SetkaT):
          shagX:=SetkaX[2]-SetkaX[1]:
          shagT:=SetkaT[2]-SetkaT[1]:

          for i from 1 to NSetkaX do
                    for j from 1 to NSetkaT do
                              UU[i, j]:=eval(eval(Pstep, x=SetkaX
[i]), t=SetkaT[j]):
```

```
                    od:
          od:

          for i from 1 to NSetkaX do
                    for j from 1 to NSetkaT do
                              Ux[i, j]:=eval(eval(GradPstep,
x=SetkaX[i]), t=SetkaT[j]):
                    od:
          od:

          for i from 1 to NSetkaX do
                    for j from 1 to NSetkaT do
                              Ut[i, j]:=eval(eval(DtPstep,
x=SetkaX[i]), t=SetkaT[j]):
                    od:
          od:

          [UU, Ux, Ut, stepPoros, coefPoros, pl0, pL0, Kosh]:
end proc:



#-------------------------------------------------------------
----
#-------------------------                                  ------
----
#-------------------------------------------------------------
----
solverOsred:=proc(stepPerm, coefPerm, stepPoros, coefPoros, l, L,
pl0, pL0, Kosh, SetkaX, SetkaT)
          local Kapprox, Mapprox, PDE, IBC, pds, g;
          local solut, NSetkaX, NSetkaT, shagX, shagT, i, j, U,
kUx, Ut;
          option remember:
          Kapprox:=(x)^stepPerm*coefPerm:
          Mapprox:=(x)^stepPoros*coefPoros:

          # ,     ,

          PDE := Mapprox*diff(u(x, t), t) = diff(Kapprox*diff(u
(x, t),x), x);
          IBC := {u(l, t) = pl0, u(x, 0) = Kosh, u(L, t) = pL0}:
          pds := pdsolve(PDE, IBC, numeric, time=t, range=l..L,
compile=true):

          pds:-value(output = listprocedure);
          solut:=subs(%[3], u(x, t)):

          #
          NSetkaX:=numelems(SetkaX):
          NSetkaT:=numelems(SetkaT):
          shagX:=SetkaX[2]-SetkaX[1]:
          shagT:=SetkaT[2]-SetkaT[1]:

          for i from 1 to NSetkaX do
                    for j from 1 to NSetkaT do
                              U[i, j]:=solut(SetkaX[i], SetkaT[j]):
```

```
                    od:
          od:

          for i from 1 to NSetkaX-1 do
                    for j from 1 to NSetkaT do
                              kUx[i, j]:=subs(x=SetkaX[i+1],
Kapprox)*(solut(SetkaX[i+1], SetkaT[j])-solut(SetkaX[i],
SetkaT[j]))/shagX:
                    od:
          od:

          for i from 1 to NSetkaX do
                    for j from 1 to NSetkaT-1 do
                              Ut[i, j]:=(solut(SetkaX[i],
SetkaT[j+1])-solut(SetkaX[i], SetkaT[j]))/shagT:
                    od:
          od:

           [U, kUx, Ut]
end proc:



#
RealDataInPerm:=proc(ANK, leftX, rightX)
          local  ak, bk, ck, GamNekolector:
          local N, i, shagX, X, Perm:

          #
          ak:=-5.58: bk:=0.45: ck:=4.61: GamNekolector:=0.6:
          N:=numelems(ANK);

          #
          for i from 1 to N do
                    Perm[i] := exp(ak*ANK[i]^2/GamNekolector^2+
bk*ANK[i]/GamNekolector+ck):
          end do:

          min([seq(Perm[k],k=1..N)]):
          max([seq(Perm[k],k=1..N)]):

          [(Perm[2])/%, seq((Perm[k])/%,k=2..N)]
end proc:



#
RealDataInPoros:=proc(ANK, leftX, rightX)
          local aphi, bphi, cphi, GamNekolector:
          local N, i, shagX, X, Poros:

          #
          aphi:=3.93: bphi:=1.47: cphi:=-0.19: GamNekolector:=0.6:
          N:=numelems(ANK);

          #
          for i from 1 to N do
```

```
                    Poros[i] := 1/(aphi+exp(bphi*ANK[i]
/GamNekolector+cphi)):
          end do:

          min([seq(Poros[k],k=1..N)]):
          max([seq(Poros[k],k=1..N)]):

          [seq((Poros[k])/%,k=1..N)]
end proc:


#        V A L U E
SetkaRavnomer:=proc(leftValue, rightValue, nValue)
          local i, shagValue, Value:

          #                    VALUE
          shagValue:=evalf((rightValue-leftValue)/nValue):
          Value[1]:=leftValue:

          for i from 2 to nValue do
                    Value[i] :=Value[i-1]+shagValue:
          end do:

          Value[nValue+1]:=rightValue;

          [seq(Value[k],k=1..nValue+1)]
end proc:


#
PieceTo1DMassive:=proc(Piece, xValue)
          local N, i, assive:

          N:=numelems(xValue):
          for i from 1 to N do
                    assive[i] :=eval(Piece, x=xValue[i]):
          end do:

          [seq(assive[k],k=1..N)]
end proc:


#
MassiveToPiece:=proc(Massive, xValue)
          local N:
          N:=numelems(Massive):
          Matrix([[seq(x>=xValue[i] and x<=xValue[i+1], i = 1 ..
N-1), x>xValue[N]], [seq(Massive[i],i=1..N-1), Massive[N]]]):

          piecewise(seq(i, i in %)):
end proc:
```

```
#                  L 2
FunPieceMinusStep:=proc(Piece, LeftStep, RightStep, NStep,
LeftCoef, RightCoef, NCoef, SetkaX)
          local i,j:
          local Coef, Step, NSetkaX, RaznicaStupStepen, Gunkzional:

          Coef:=SetkaRavnomer(LeftCoef, RightCoef, NCoef);

          Step:=SetkaRavnomer(LeftStep, RightStep, NStep);

          NSetkaX:=numelems(SetkaX):

          for i from 1 to NStep+1 do
                    RaznicaStupStepen:=Piece-BB*x^AA:
                    for j from 1 to NCoef+1 do
                              subs(BB=Coef[j],AA=Step[i],
RaznicaStupStepen)^2:
                              Gunkzional[i,j]:=(add(eval(%, x =
SetkaX[hh])*(SetkaX[hh]-SetkaX[hh-1]),hh=2..NSetkaX)):
                    od:
          od:

          seq([seq([Step[i], Coef[j], Gunkzional[i,j]], i = 1 ..
NStep+1)], j = 1 .. NCoef+1)
end proc:



#
minimizeFun:=proc(Fun)
          local i, j ,NCoef, NStep, dddd, NomerMin:

          NStep:=numelems(Fun):
          NCoef:=numelems(Fun[1]):

          dddd:=Matrix([seq([seq(Fun[i,j,3], i = 1 .. NStep)], j
= 1 .. NCoef)]);
          NomerMin:=min[index](dddd);

          [Fun[NomerMin[2],NomerMin[1],1],Fun[NomerMin[2],
NomerMin[1],2],Fun[NomerMin[2],NomerMin[1],3]]:
          evalf(%);
end proc:



#
maximizeFun:=proc(Fun)
          local i, j ,NCoef, NStep, dddd, NomerMax:

          NStep:=numelems(Fun):
          NCoef:=numelems(Fun[1]):

          dddd:=Matrix([seq([seq(Fun[i,j,3], i = 1 .. NStep)], j
= 1 .. NCoef)]);
          NomerMax:=max[index](dddd);
```

```
          [Fun[NomerMax[2],NomerMax[1],1],Fun[NomerMax[2],
NomerMax[1],2],Fun[NomerMax[2],NomerMax[1],3]]:
          evalf(%);
end proc:



#
FunPressPieceMinusStep:=proc(LeftStep, RightStep, NStep,
LeftCoef, RightCoef, NCoef, SetkaX, SetkaT, ReshStupen)
          local i,j:
          local Coef, Step, NSetkaT, shagT, NSetkaX, shagX,
RaznicaStupStepen, Funkzional:
          local U1, U1x, U1t, U2, U2x, U2t:
          local stepPoros, coefPoros, l, L, pl0, pL0, Kosh:
          local rr, kk, ReshPower, hh, IntSol_x, IntSol_xt, pp:

          Coef:=SetkaRavnomer(LeftCoef, RightCoef, NCoef);

          Step:=SetkaRavnomer(LeftStep, RightStep, NStep);

          NSetkaX:=numelems(SetkaX):
          NSetkaT:=numelems(SetkaT):
          shagX:=SetkaX[2]-SetkaX[1]:
          shagT:=SetkaT[2]-SetkaT[1]:

          #
          l:=SetkaX[1]:
          L:=SetkaX[NSetkaX]:

          U1:=ReshStupen[1]:
          U1x:=ReshStupen[2]:
          U1t:=ReshStupen[3]:

          stepPoros:=ReshStupen[4]:
          coefPoros:=ReshStupen[5]:

          #            -

          pl0:=ReshStupen[6]:
          pL0:=ReshStupen[7]:
          Kosh:=ReshStupen[8]:



          #
          for rr from 1 to NStep+1 do
                    for kk from 1 to NCoef+1 do
                              IntSol_xt:=0:

                              ReshPower:=solverOsred(Step[rr],
Coef[kk], stepPoros, coefPoros, l, L, pl0, pL0, Kosh, SetkaX,
SetkaT):
                              U2:=ReshPower[1]:
                              U2x:=ReshPower[2]:
                              U2t:=ReshPower[3]:
                              for hh from 1 to NSetkaT-1 do
```

```
                                        IntSol_x:=0:
                                        for pp from 1 to
NSetkaX-1 do
                                        IntSol_x:=IntSol_x+((U1
[pp, hh]-U2[pp, hh])^2+(U1x[pp, hh]-U2x[pp, hh])^2+(U1t[pp, hh]-
U2t[pp, hh])^2)*shagX:
                                        end do:
                                        IntSol_xt:=IntSol_x+
IntSol_x*shagT:
                              end do:
                              Funkzional[rr,kk]:=IntSol_xt;
p r i n t f ( " = % f ,   = % f ,                  =%f \n", Step[rr],
Coef[kk], %):
                    end do:
                    print(NStep+1-rr):
          end do:

          seq([seq([Step[i], Coef[j], Funkzional[i,j]], i = 1 ..
NStep-1)], j = 1 .. NCoef-1)
end proc:


#
SetkiPlusOptimalStepsForPermAndPoros:=proc(RealData, l, L)
          local NN, SetkaX, SetkaT, xDiscont, PorosMassive,
PorosPiece, PermMassive, PermPiece, FF:
          local stepPorosL2, coefPorosL2, stepPermL2, coefPermL2:
          local plotPowerPoros, plotPowerPerm:


          #
          NN:=numelems(RealData):

          #       x   (   ,     )
          SetkaX:=SetkaRavnomer(l, L, 3*NN):

          #     ( ,       )

          xDiscont:=SetkaRavnomer(l, L, NN):

          #
(  -   )
          PorosMassive:=RealDataInPoros(RealData6508_1269_1274, l, L):
          PorosPiece:=MassiveToPiece(PorosMassive,xDiscont):
          PermMassive:=RealDataInPerm(RealData6508_1269_1274, l, L):
          PermPiece:=MassiveToPiece(PermMassive,xDiscont):

          #
          p r i n t f ( "   -
     ( )   \ n                          ");
          FF:=FunPieceMinusStep(PorosPiece, 0, 2, 50, 0, 2, 50,
SetkaX):
          minimizeFun([FF]):
          stepPorosL2:=%[1]:
          coefPorosL2:=%%[2]:
          p r i n t f ( "        =%f,   = % f   \ n " ,
stepPorosL2, coefPorosL2):
```

```
          plotPowerPoros:=plot([coefPorosL2*x^stepPorosL2,
PorosPiece], x=l..L, color=[black, red], labelfont = ["Verdana",
bold, 14], labels=[x, m], thickness=2);

          #
          p r i n t f ( "   -
     ( )   \ n                              ");
          FF:=FunPieceMinusStep(PermPiece, 0, 3, 50, 0, 2, 50,
SetkaX):
          minimizeFun([FF]):
          stepPermL2:=%[1]:
          coefPermL2:=%%[2]:
          p r i n t f ( "        =%f,                  =%f \n",
stepPermL2, coefPermL2):
          plotPowerPerm:=plot([coefPermL2*x^stepPermL2,
PermPiece], x=l..L, color=[black, red], labelfont = ["Verdana",
bold, 14], labels=[x, k], thickness=2);

[stepPermL2, coefPermL2, stepPorosL2, coefPorosL2, PermMassive,
xDiscont, SetkaX, plotPowerPoros, plotPowerPerm]:
end proc:

DisplayFun:=proc(F)
          local MinFun, Fplot, pointFminplot:
          MinFun:=minimizeFun([F]);

          Fplot:=surfdata([funkcional6508_1269_1274], labels =
[b, a, "F(a,b)/Fmin"], labelfont = ["Verdana", bold, 14],
shading=none, axes=boxed):

          pointFminplot:=pointplot3d(Fmin, color = black, axes =
boxed, symbol = solidsphere, symbolsize=20, filled=true,
transparency = .7):

Fplot, pointFminplot;
end proc:
> #

#
RealData6508_1269_1274:=[.557, .631, .614, .619, .623, .570,
.464, .485, .536, .522, .576, .518, .507, .485, .481, .461, .511,
.523, .493, .436, .436, .441, .495, .464, .376, .398, .474, .484,
.471]:

RealData6508_1269_1274:=[.557, .631, .614, .619, .623, .570,
.464]:
l:=0: L:=1:

SetkiPlusOptimalStepsForPermAndPoros6508_1269_1274:=
SetkiPlusOptimalStepsForPermAndPoros(RealData6508_1269_1274, l,
L);
stepPermL2:=SetkiPlusOptimalStepsForPermAndPoros6508_1269_1274[1]
:
coefPermL2:=SetkiPlusOptimalStepsForPermAndPoros6508_1269_1274[2]
:
stepPorosL2:=SetkiPlusOptimalStepsForPermAndPoros6508_1269_1274
```

```
[3]:
coefPorosL2:=SetkiPlusOptimalStepsForPermAndPoros6508_1269_1274
[4]:
PermMassive:=SetkiPlusOptimalStepsForPermAndPoros6508_1269_1274
[5]:
xDiscont:=SetkiPlusOptimalStepsForPermAndPoros6508_1269_1274[6]:
SetkaX:=SetkiPlusOptimalStepsForPermAndPoros6508_1269_1274[7]:
plotPowerPoros:=
SetkiPlusOptimalStepsForPermAndPoros6508_1269_1274[8]:
plotPowerPerm:=SetkiPlusOptimalStepsForPermAndPoros6508_1269_1274
[9]:



#------------------------------------------------------------------
------
```