```
> with(plots):Digits:=100:interface(displayprecision=10):with
  (linalg):


> N:=5;
  KK:=4;
  ### Change of notation: Now the indices K[i] are in order not
  alpha's
  # vector U shows  if the branch is up (1) or down (0)
  U:=vector(N,[]):
  alpha:=vector(N,[]):
  K:=vector(N,[]):
  a:=vector(N,[]):
  bb:=vector(N+1,[]):
  c:=vector(KK,[]):
  for j from 1 to N do
  U[j]:=0;
  od:
  alpha[1]:=0.5:K[1]:=1:U[K[1]]:=0:
  alpha[2]:=0.3:K[2]:=2:U[K[2]]:=0:#
  alpha[3]:=0.3:K[3]:=4:U[K[3]]:=1:  #
  alpha[4]:=0.5:K[4]:=5:U[K[4]]:=1:
  i:='i':

  beta:=N-KK+sum(alpha[i],i=1..KK);i:='i';
  delta1:=(xw,yw)-> piecewise(xw<=yw,0,1):
  for j from 1 to N do
  b[j]:=(j-1-sum((1-alpha[i])*delta1( j,K[i]) ,i=1..KK))/beta;
  bb[j]:=b[j];
   od:  i:='i':
  b[N+1]:=1:  bb[N+1]:=1:
  for j from 1 to N do
  a[j]:=(j-1-sum((1-alpha[i])*delta1( j,K[i]-U[j]) ,i=1..KK));
   od:


  for j from 1 to KK do
  if U[K[j]]=0 then   c[j]:=b[K[j]+1];
               else  c[j]:=b[K[j]];fi;
   #print(`c[`,j,`] = `,c[j]) ;
  od:
  print(`alpha = `,alpha);
  print(`K = `,K);
  print(`b = `,bb);
  print(`a = `,a);
  print(`c = `,c);
```

```
>
   maa:=a[2]-a[1]:# maximum a[i+1]-a[i]
   for i from 3 to N do
   if (a[i]-a[i-1])>maa then maa:=(a[i]-a[i-1]) fi
   od;#
   maa;
   beta_max:=evalf(1+(a[N]-a[1])/maa);
> if beta> beta_max then print("ERROR") fi;


>
   uint_of_x:=x->piecewise(x<b[2],1,# This function needs additions
   by hand for
                                    # N>9 . Automathic procedure
   causes plotting problems
                                    # but is used in other
   programs
                           x<b[3],2,
                           x<b[4],3,
                           x<b[5],4,
                           x<b[6],5,
                           x<b[7],6,
                           x<b[8],7,
                           x<b[9],8,
                                9);
   int_of_x:=x->piecewise(x<=b[2],1,# This function needs additions
   by hand for
                                    # N>9 . Automathic procedure
   causes plotting problems
                                    # but is used in other
   programs
                           x<=b[3],2,
                           x<=b[4],3,
                           x<=b[5],4,
                           x<=b[6],5,
                           x<=b[7],6,
                           x<=b[8],7,
                           x<=b[9],8,
                                9);
   x:='x':
   uT:=x->beta*x-a[uint_of_x(x)];
   T:=x->beta*x-a[int_of_x(x)];
   for j from 1 to KK do
   if U[K[j]]=0 then    Tc:=T(c[j]);
                else   Tc:=uT(c[j])fi;
   print(`T(c[`,j,`]) =`, Tc)
   od;
```

```
plot([uT(x), x, 0, 1, 1-alpha[1], 1-alpha[2]], x=0..1, thickness=[2, 1, 1, 1, 1, 1]);
plot([T(x), x, 0, 1, alpha[1], alpha[2]], x=0..1, thickness=[2, 1, 1, 1, 1, 1]);
```

$$N := 5$$

$$KK := 4$$

$$\beta := 2.6000000000$$

$$i := i$$

$$alpha = , \left[\begin{array}{ccccc} 0.5000000000 & 0.3000000000 & 0.3000000000 & 0.5000000000 & \alpha_5 \end{array} \right]$$

$$K = , \left[\begin{array}{ccccc} 1 & 2 & 4 & 5 & K_5 \end{array} \right]$$

$$b = , \left[\begin{array}{cccccc} 0.0000000000 & 0.1923076923 & 0.3076923077 & 0.6923076923 & 0.8076923077 & 1 \end{array} \right]$$

$$a = , \left[\begin{array}{ccccc} 0.0000000000 & 0.5000000000 & 0.8000000000 & 1.1000000000 & 1.6000000000 \end{array} \right]$$

$$c = , \left[\begin{array}{cccc} 0.1923076923 & 0.3076923077 & 0.6923076923 & 0.8076923077 \end{array} \right]$$

$$0.5000000000$$

$$\beta max := 4.2000000000$$

$$uint\_of\_x := x \rightarrow piecewise\left(x < b_2, 1, x < b_3, 2, x < b_4, 3, x < b_5, 4, x < b_6, 5, x < b_7, 6, x < b_8, 7, x < b_9, 8, 9\right)$$

$$int\_of\_x := x \rightarrow piecewise\left(x \leq b_2, 1, x \leq b_3, 2, x \leq b_4, 3, x \leq b_5, 4, x \leq b_6, 5, x \leq b_7, 6, x \leq b_8, 7, x \leq b_9, 8, 9\right)$$

$$uT := x \rightarrow \beta x - a_{uint\_of\_x(x)}$$

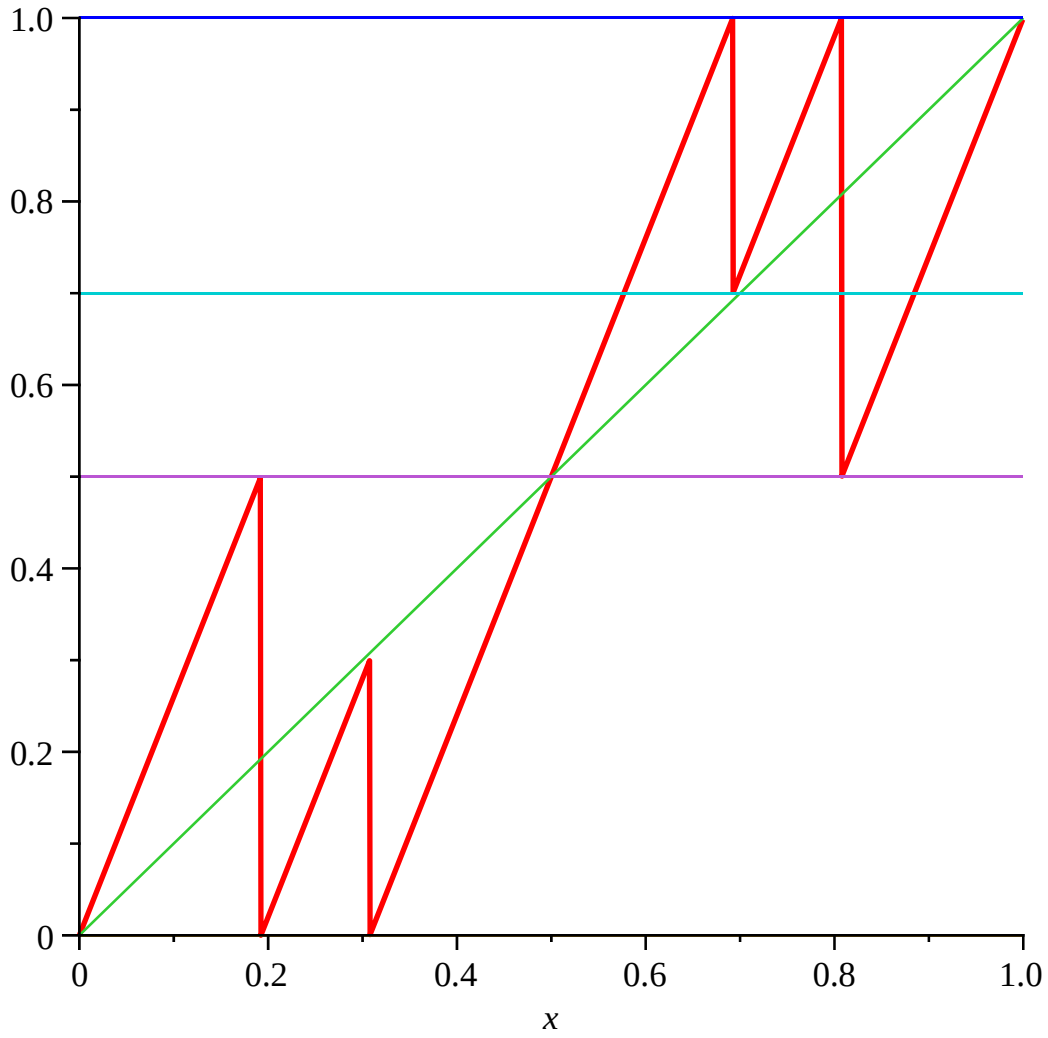
$$T := x \rightarrow \beta x - a_{int\_of\_x(x)}$$

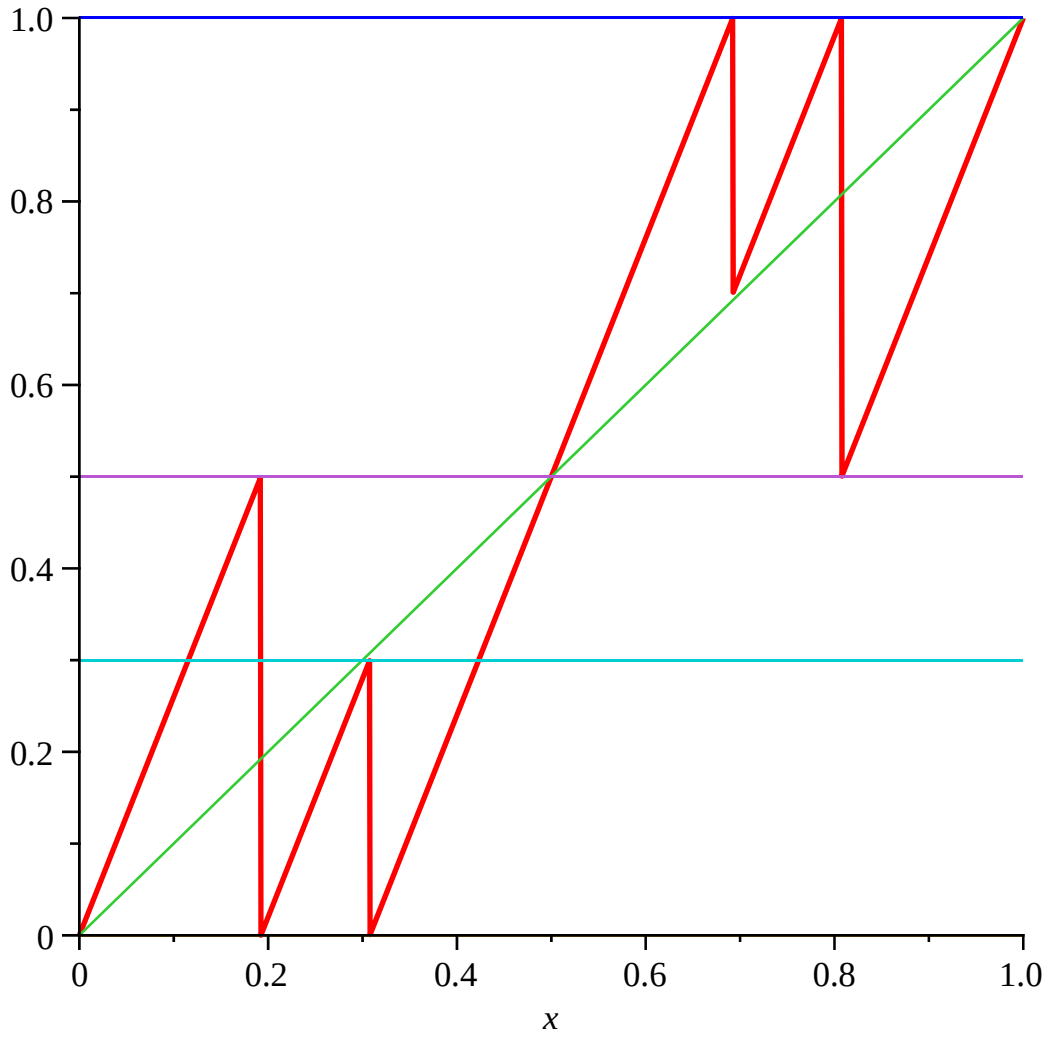
$$T(c[, 1,]) =, 0.5000000000$$

$$T(c[, 2,]) =, 0.3000000000$$

$$T(c[, 3,]) =, 0.7000000000$$

$$T(c[, 4,]) =, 0.5000000000$$


1.0
0.8
0.6
0.4
0.2
0
0
0.2
0.4
0.6
0.8
1.0
x




```
>
> ud:=vector(50): Digits:=100; NN:=50;
  d:=vector(50):
  xx:=evalf(rand()/10^12);
  xxt:=xx:
  for i from 1 to NN do
  ud[i]:=a[uint_of_x(xxt)];
  xxt:=uT(xxt);
  od:
  xxt:=xx:
  for i from 1 to NN do
  d[i]:=a[int_of_x(xxt)];
  xxt:=T(xxt);
  od:
  print(ud);
  uls_it_x:=evalf(sum(ud[j1]/beta^j1, j1=1..NN));
  print(d);
  ls_it_x:=evalf(sum(d[j1]/beta^j1, j1=1..NN));
  terr:=xx-uls_it_x;
  err:=xx-ls_it_x;
```

$$Digits := 100$$

$$NN := 50$$

$$xx := 0.3957188605$$

[0.8000000000, 0.5000000000, 0.0000000000, 0.5000000000, 0.0000000000, 0.8000000000,
0.0000000000, 0.8000000000, 0.8000000000, 0.0000000000, 0.0000000000,
0.0000000000, 0.0000000000, 0.0000000000, 0.5000000000, 0.5000000000,
0.0000000000, 0.0000000000, 0.8000000000, 0.0000000000, 0.5000000000,
0.0000000000, 0.0000000000, 0.8000000000, 0.8000000000, 0.0000000000,
0.8000000000, 0.8000000000, 0.8000000000, 0.0000000000, 0.8000000000,
0.5000000000, 0.0000000000, 0.8000000000, 0.8000000000, 0.0000000000,
0.5000000000, 0.5000000000, 0.0000000000, 0.8000000000, 0.8000000000,
0.8000000000, 0.8000000000, 0.0000000000, 0.8000000000, 0.0000000000,
0.0000000000, 0.8000000000, 0.8000000000, 0.0000000000]

$$uIs\_it\_x := 0.3957188605$$

[0.8000000000, 0.5000000000, 0.0000000000, 0.5000000000, 0.0000000000, 0.8000000000,
0.0000000000, 0.8000000000, 0.8000000000, 0.0000000000, 0.0000000000,
0.0000000000, 0.0000000000, 0.0000000000, 0.5000000000, 0.5000000000,
0.0000000000, 0.0000000000, 0.8000000000, 0.0000000000, 0.5000000000,
0.0000000000, 0.0000000000, 0.8000000000, 0.8000000000, 0.0000000000,
0.8000000000, 0.8000000000, 0.8000000000, 0.0000000000, 0.8000000000,
0.5000000000, 0.0000000000, 0.8000000000, 0.8000000000, 0.0000000000,
0.5000000000, 0.5000000000, 0.0000000000, 0.8000000000, 0.8000000000,
0.8000000000, 0.8000000000, 0.0000000000, 0.8000000000, 0.0000000000,
0.0000000000, 0.8000000000, 0.8000000000, 0.0000000000]

$$Is\_it\_x := 0.3957188605$$

$$terr := 5.758853785\ 10^{-23}$$

$$err := 5.758853785\ 10^{-23} \quad \textbf{(1)}$$

```
>
> NN:=50; chi:=(x1,x2,t)->piecewise(t<x1,0,t<=x2,1,0);
  uchi:=(x1,x2,t)->piecewise(t<x1,0,t<x2,1,0);

  #Expansion of c1, c2 ... and all the S's

  for i from 1 to KK do
  xxt:=c[i]; upflag:=U[K[i]];
       for n from 1 to NN+1 do
        if upflag=1 then intx:=uint_of_x(xxt) else intx:=int_of_x
  (xxt) fi;
          dc[i,n]:=a[intx];
          ic[i,n]:=intx-1;
         if upflag=0 then
              for ii from 1 to KK do
                   if xxt>c[ii] then cc[i,ii,n]:=1 else cc[i,ii,n]
  :=0 fi;
```

```
            od;
        fi;
        if upflag=1 then
            for ii from 1 to KK do
                if xxt<c[ii] then cc[i,ii,n]:=1 else cc[i,ii,n]
:=0 fi;
            od;
    fi;
    valc[i,n]:=xxt;
    if upflag=0 then xxt:=T(xxt) else xxt:=uT(xxt)fi;
    od:
Is_it_x:=sum(dc[i,j1]/beta^j1,j1=1..NN);
S[i]:=sum(ic[i,j1+1]/beta^(j1+1),j1=1..NN);
od;
for i from 1 to KK do
for j from 1 to KK do
SS[i,j]:=sum(cc[i,j,j1+1]/beta^(j1+1),j1=1..NN);

#print(`SS[`,i,j,`] =`,SS[i,j]);
od;od;
#for i from 1 to 30 do
#print(cc[2,1,i],cc[2,2,i]) od;
```

$$NN := 50$$

$$\chi := (x1, x2, t) \rightarrow piecewise(t < x1, 0, t \leq x2, 1, 0)$$

$$uchi := (x1, x2, t) \rightarrow piecewise(t < x1, 0, t < x2, 1, 0)$$

$$xxt := 0.1923076923$$

$$upflag := 0$$

$$Is\_it\_x := 0.1923076923$$

$$S_1 := 0.4807692308$$

$$xxt := 0.3076923077$$

$$upflag := 0$$

$$Is\_it\_x := 0.3076923077$$

$$S_2 := 0.2309716887$$

$$xxt := 0.6923076923$$

$$upflag := 1$$

$$Is\_it\_x := 0.6923076923$$

$$S_3 := 0.7305667729$$

$$xxt := 0.8076923077$$

$$upflag := 1$$

$$Is\_it\_x := 0.8076923077$$

$$S_4 := 0.4807692308$$

```
>
MM:=matrix(KK,KK,[]):
for i from 1 to KK do
for j from 1 to KK do

MM[j,i]:=-SS[i,j];
od;od;
print(`MM = `,MM);
print(`1/beta =`, 1/beta);

print(`eigenvalues of  -S =`,eigenvalues(MM));

ve:=vector(KK,[]):
for i from 1 to KK do
ve[i]:=1/beta;

MM[i,i]:=MM[i,i]+1/beta;
od:
det(MM);
print(MM);
print(ve);
print(`1/beta(beta-1)  =`, 1/(beta*(beta-1)));



DD:=linsolve(MM,ve);
for i from 1 to 10 do
print(i,1/(beta^i*(beta-1)));od;
SS[2,1]/SS[2,2];
```

$$MM = , \begin{bmatrix} -0.2403846154 & -0.2304655439 & -0.0000000000 & -0.0000000000 \\ -0.2403846154 & -0.0005061447 & -0.0000000000 & -0.0000000000 \\ -0.0000000000 & -0.0000000000 & -0.0005061447 & -0.2403846154 \\ -0.0000000000 & -0.0000000000 & -0.2304655439 & -0.2403846154 \end{bmatrix}$$

$$1/beta =, 0.3846153846$$

$$eigenvalues\ of\ -S =, -0.3846153846, 0.1437246245, 0.1437246245, -0.3846153846$$

$$1.760892889\ 10^{-43}$$

$$\begin{bmatrix} 0.1442307692 & -0.2304655439 & -0.0000000000 & -0.0000000000 \\ -0.2403846154 & 0.3841092399 & -0.0000000000 & -0.0000000000 \\ -0.0000000000 & -0.0000000000 & 0.3841092399 & -0.2403846154 \\ -0.0000000000 & -0.0000000000 & -0.2304655439 & 0.1442307692 \end{bmatrix}$$

$$\begin{bmatrix} 0.3846153846 & 0.3846153846 & 0.3846153846 & 0.3846153846 \end{bmatrix}$$

1/beta(beta-1) =, 0.2403846154

$$DD := \left[5.632937327\ 10^{20} \quad 3.525224942\ 10^{20} \quad 3.525224942\ 10^{20} \quad 5.632937327\ 10^{20} \right]$$

1, 0.2403846154

2, 0.0924556213

3, 0.0355598543

4, 0.0136768671

5, 0.0052603335

6, 0.0020232052

7, 0.0007781558

8, 0.0002992907

9, 0.0001151118

10, 0.0000442738

455.3352528385 **(2)**

>

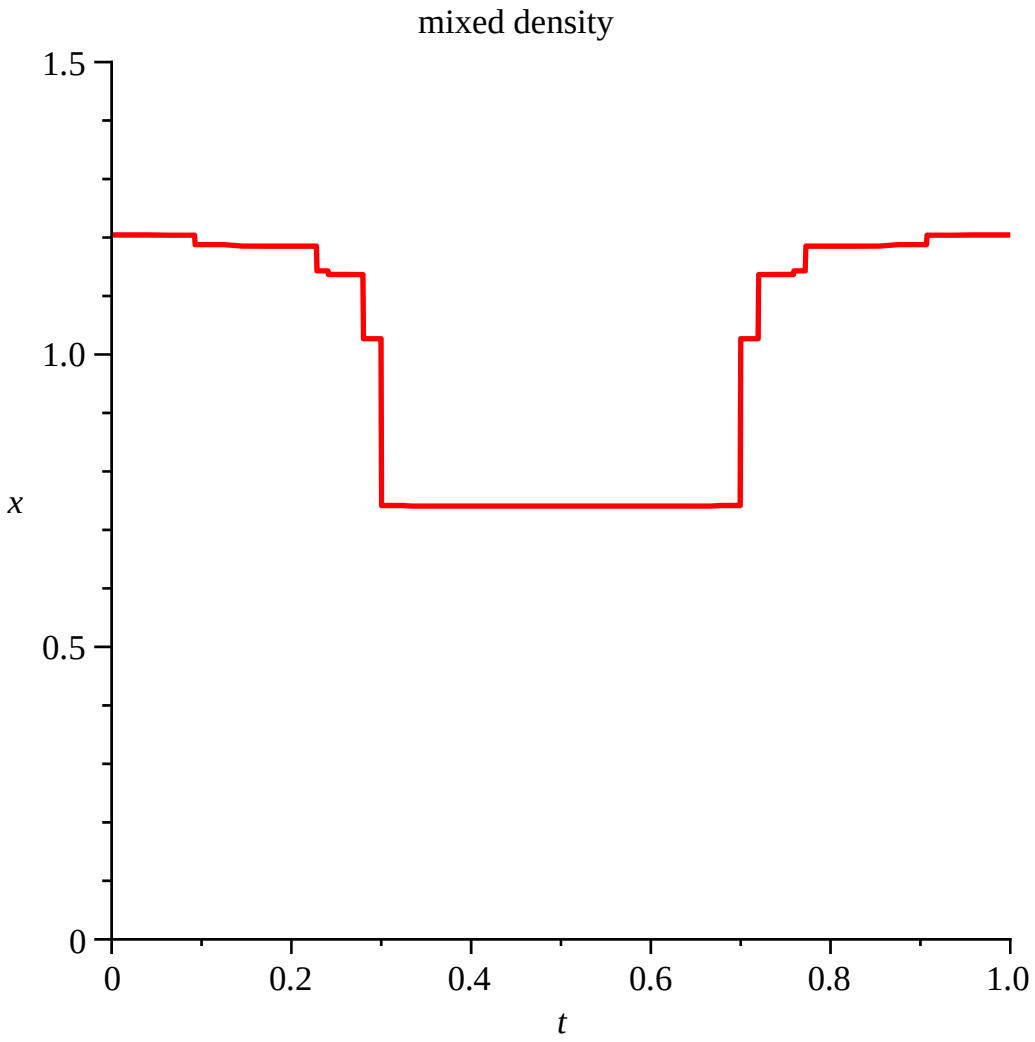
```
density:=proc(t) local j, den;
                  den:=1/beta;
                 for j from 1 to KK do
                 if U[K[j]]=0 then
                   den:=den+ DD[j]*sum((chi(0,valc[j,i1+1],t))
/beta^(i1+1),i1=1..50)
                                    else
                   den:=den+ DD[j]*sum((uchi(valc[j,i1+1],1,t))
/beta^(i1+1),i1=1..50)
                  fi;
                  od;
                 return den;
          end proc;
#Normalizing factor
tNC:=1/beta:
for j from 1 to KK do
 tNC:=tNC+tDD[j]*sum((1-tvalc[j,i1+1])/beta^(i1+1),i1=1..50)
 od:
NC:=1/beta:
for j from 1 to KK do
if U[K[j]]=0 then
 NC:=NC+DD[j]*sum((valc[j,i1+1])/beta^(i1+1),i1=1..50)
else
NC:=NC+DD[j]*sum((1-valc[j,i1+1])/beta^(i1+1),i1=1..50)
fi;
 od:

print(`NC = `,NC);

plot([(1/NC)*density(t)],t=0..1-0.00000000001,x=0..1.5,title=
```

```
    "mixed density", thickness=2);
density := proc(t)
    local j, den;
    den := 1/beta;
    for j to KK do
        if U[K[j]] = 0 then
            den := den + DD[j]*(sum(chi(0, valc[j, i1+1], t)/beta^(i1+1), i1=1..50))
        else
            den := den + DD[j]*(sum(uchi(valc[j, i1+1], 1, t)/beta^(i1+1), i1=1..50))
        end if
    end do;
    return den
end proc
```

$$NC = , 1.827969023\ 10^{20}$$




```
>
>
    #check density greedy
    #preimages
    for j6 from 1 to KK-1 do
```

```
y[j6]:=alpha[j6]+(alpha[j6+1]-alpha[j6])*rand()/10^12;
od:
y[0]:=alpha[1]*rand()/10^12:
y[KK]:=alpha[KK]+(1-alpha[KK])*rand()/10^12;
for j6 from 0 to KK do
for i3 from 1 to N do
pre[i3]:=(y[j6]+a[i3])/beta;
od;
#plot([T(t),0,1,y[j6],tT(tc[1]),tT(tc[2])],t=0..1,color=[red,
black,black,green,yellow,yellow]);
su:=0:
for i3 from 1 to N do
if (pre[i3]>=b[i3] and pre[i3]<=b[i3+1]) then
 su:=su+density(pre[i3])/beta;
print(i3);
fi;
od;
err[j6]:=density(y[j6])-su;
od;

for j6 from 0 to KK do
print(`y =`,y[j6]);
print(`err[`,j6,`]=`,err[j6]);
od;
```

$$y_4 := 0.9213113422$$

$$su := 0$$

1

2

3

$$err_0 := -0.0049677495$$

$$su := 0$$

1

3

$$err_1 := -0.0049677495$$

$$su := 0$$

1

2

3

$$err_2 := -0.0049677495$$

$$su := 0$$

1

3

$$err_3 := -0.0049677495$$

$$su := 0$$

3

4

5

$$err_4 := -0.0049677495$$

$$y =, 0.2137760284$$

$$err[, 0,]=, -0.0049677495$$

$$y =, 0.4613720367$$

$$err[, 1,]=, -0.0049677495$$

$$y =, 0.3000000000$$

$$err[, 2,]=, -0.0049677495$$

$$y =, 0.4600374969$$

$$err[, 3,]=, -0.0049677495$$

$$y =, 0.9213113422$$

$$err[, 4,]=, -0.0049677495$$

>

```
#check density greedy
#preimages
y:=0.500000000000;
for i2 from 1 to N do
pre[i2]:=(y+a[i2])/beta;
od;
plot([T(t),0,1,y,T(c[1]),T(c[2])],t=0..1,color=[red,black,black,
green,yellow,yellow]);
su:=0:
for i2 from 1 to N do
if (pre[i2]>=b[i2] and pre[i2]<=b[i2+1]) then
 su:=su+density(pre[i2])/beta;
print(i2);
fi;
od;
err2:=density(y)-su;
```

$$y := 0.5000000000$$

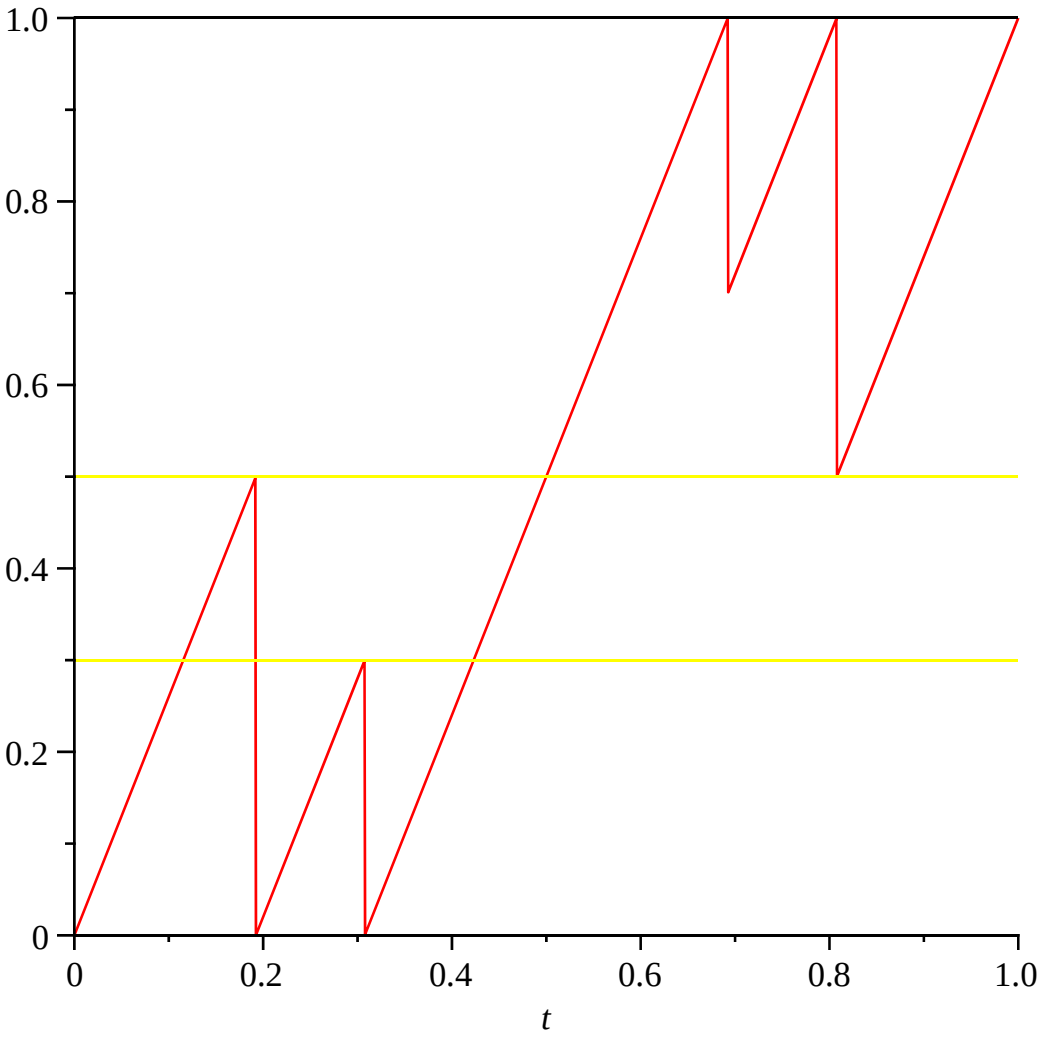
$$pre_1 := 0.1923076923$$

$$pre_2 := 0.3846153846$$

$$pre_3 := 0.5000000000$$

$$pre_4 := 0.6153846154$$

$$pre_5 := 0.8076923077$$



1

3

5

$err2 := -0.0605834296$

>

>