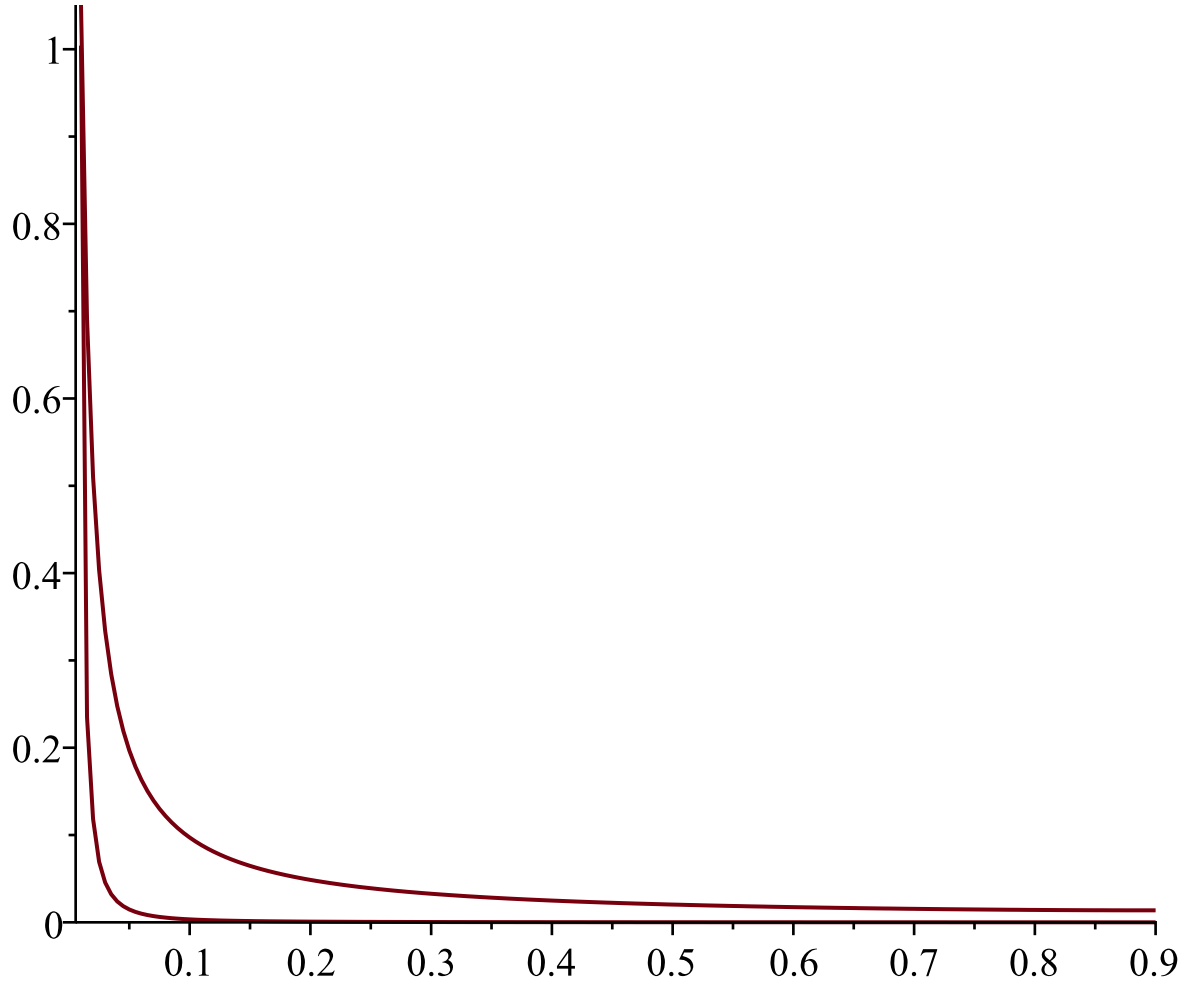
```
> restart:
  with(plots):
  with(LinearAlgebra):
  #-------------------------------------------------------------------
  -
  # ( ! )
  # ( ! ) ,              ,

  # ( ! )
    (       )
  #-------------------------------------------------------------------
  -

  #
  r0:=0.00000001: #      (   0   )
  R:=1: #
  alpha:=evalf(1/4): #
  Pn1:=100: #
    (         )
  Pright:=0: #

  #
  powFrac:=proc(chislo,stepen):
           evalf(sign(chislo)*abs(chislo)^stepen):
  end proc:

  #-------------------------------------------------------------------
  -
  #--------------------                          --------------------
  -
  #-------------------------------------------------------------------
  -
  S o l v e r : =  p r o c ( n X , K , n 1 )  #  -    ,  ,

           global r0,R,alpha,Pn1,Pright:
           local RightX, LeftX, shagX, X, obshCoeff, a, b, A,
  Achert, B, E, SystemMatrix, Bright, Solution, p, c:
           local i,n,k:

           #
           RightX:=R:
           LeftX:=r0:
           X[0]:=LeftX:
           shagX:=evalf((RightX-LeftX)/nX):
           for i from 1 to nX do
                    X[i] := evalf(X[i-1]+shagX):
           end do:

           #
           p[nX]:=Pright:
           p[n1]:=Pn1:
           print(X[n1]);

  #             r ^ 2
  x
  obshCoeff:=powFrac(X[1]-X[0],alpha/2)/GAMMA(alpha/2+2):
```

```
for n from 0 to nX do
          a[0,n]:=obshCoeff*(powFrac(n-1,alpha/2+1)-(n-1-alpha/2)
*powFrac(n,alpha/2)):
          for k from 1 to n-1 do
                    a[k,n]:=obshCoeff*(powFrac(n-k+1,alpha/2+1)
                    +powFrac(n-k-1,alpha/2+1)-2*powFrac(n-k,
alpha/2+1)):
          end do:
          a[n,n]:=obshCoeff*1:
end do:

#              x
R^2
obshCoeff:=powFrac(X[1]-X[0],alpha/2)/GAMMA(alpha/2+2):
for n from 0 to nX do
          b[n,n]:=obshCoeff*1:
          for k from n+1 to nX-1 do
                    b[k,n]:=obshCoeff*(powFrac(k+1-n,alpha/2+1)
                    +powFrac(k-1-n,alpha/2+1)-2*powFrac(k-n,
alpha/2+1)):
          end do:
          b[nX,n]:=obshCoeff*(powFrac(nX-n,alpha/2)*(alpha/2+n-
nX+1)+powFrac(nX-1-n,alpha/2+1)):
end do:

#
for i from 0 to nX do
          for n from 0 to nX do
                    if i>n then
                              A[i,n]:=0;
                    fi:
                    if i<=n then
                              A[i,n]:=2^(-alpha)*a[i,n]*add(b[j,
i],j=i..nX)*X[i]^(-alpha/2);
                    fi:
          end do:
end do:

#
for n from 0 to nX do
          Achert[0,n]:=-2*sqrt(X[0])*A[0,n]/(X[1]-X[0]);
          for i from 1 to nX-1 do
                    Achert[i,n]:=2*(sqrt(X[i-1])*A[i-1,n]-sqrt(X
[i])*A[i,n])/(X[i+1]-X[i]);
          end do:
end do:


#

for n from 0 to nX do
          if n=0 then
                    B[0]:=0:
          else
                    B[n]:=1/(K*sqrt(X[n]));
          fi:
          #E[n]:=-2*sqrt(X[nX-1])*A[nX-1,n]/(X[nX]-X[nX-1])*p[nX]-
```

```
          #2*(sqrt(X[n1-1])*A[n1-1,n]-sqrt(X[n1])*A[n1,n])/(X
[n1+1]-X[n1])*p[n1]:
          E[n]:=-2*sqrt(X[nX-1])*A[nX-1,n]/(X[nX]-X[nX-1])*p[nX]-
add(Achert[i,n]*p[n1],i=0..n1):
end do:


#
SystemMatrix:=evalf(Matrix([seq([seq(Achert[i,n],i=n1+1..nX-1),-B
[n]],n=n1..nX-1)]));
Bright := Vector([seq(E[n],n=n1..nX-1)]);
Solution:=LinearSolve(SystemMatrix, Bright);



#
for i from 1 to nX-1-n1 do
          p[i]:=Solution[i];
end do:
#c:=Solution[nX]:


[X[n1],Pn1],seq([X[i+n1],p[i]],i=1..nX-1-n1);
end proc:


>
SolN100:=[Solver(200,0.8,1)]:
SolN200:=[Solver(400,0.8,2)]:
SolN400:=[Solver(800,0.8,4)]:

#[[X[1],Pn1],seq([X[i],p[i]],i=2..nX)]


#display(Nx100and200);
                          0.005000099500
                          0.005000099500
                          0.005000099500                                (1)
> seq([SolN100[i,1],(SolN100[i-1,2]-2*SolN100[i,2]+SolN100[i+1,2])/
(SolN100[1,1]-SolN100[2,1])^2/490000],i=2..180):
q1:=plot([%],linestyle = [dot, dash, solid],thickness=2,color=
black,labels=['x', P('x')],labelfont = ["Verdana",bold, 14],
legend=["n=200","n=400","n=800"],legendstyle = [font =
["Verdana",bold, 12]]);

seq([SolN100[i,1],(SolN100[i,2]-SolN100[i+1,2])/(SolN100[2,1]-
SolN100[1,1])/1600],i=2..180):
q2:=plot([%],linestyle = [dot, dash, solid],thickness=2,color=
black,labels=['x', P('x')],labelfont = ["Verdana",bold, 14],
legend=["n=200","n=400","n=800"],legendstyle = [font =
["Verdana",bold, 12]]);

display(q1,q2);
#interp([seq(SolN100[i,1],i=1..199)], [seq(SolN100[i,2],i=1..199)
], z); plot(%,z=0..1)
```

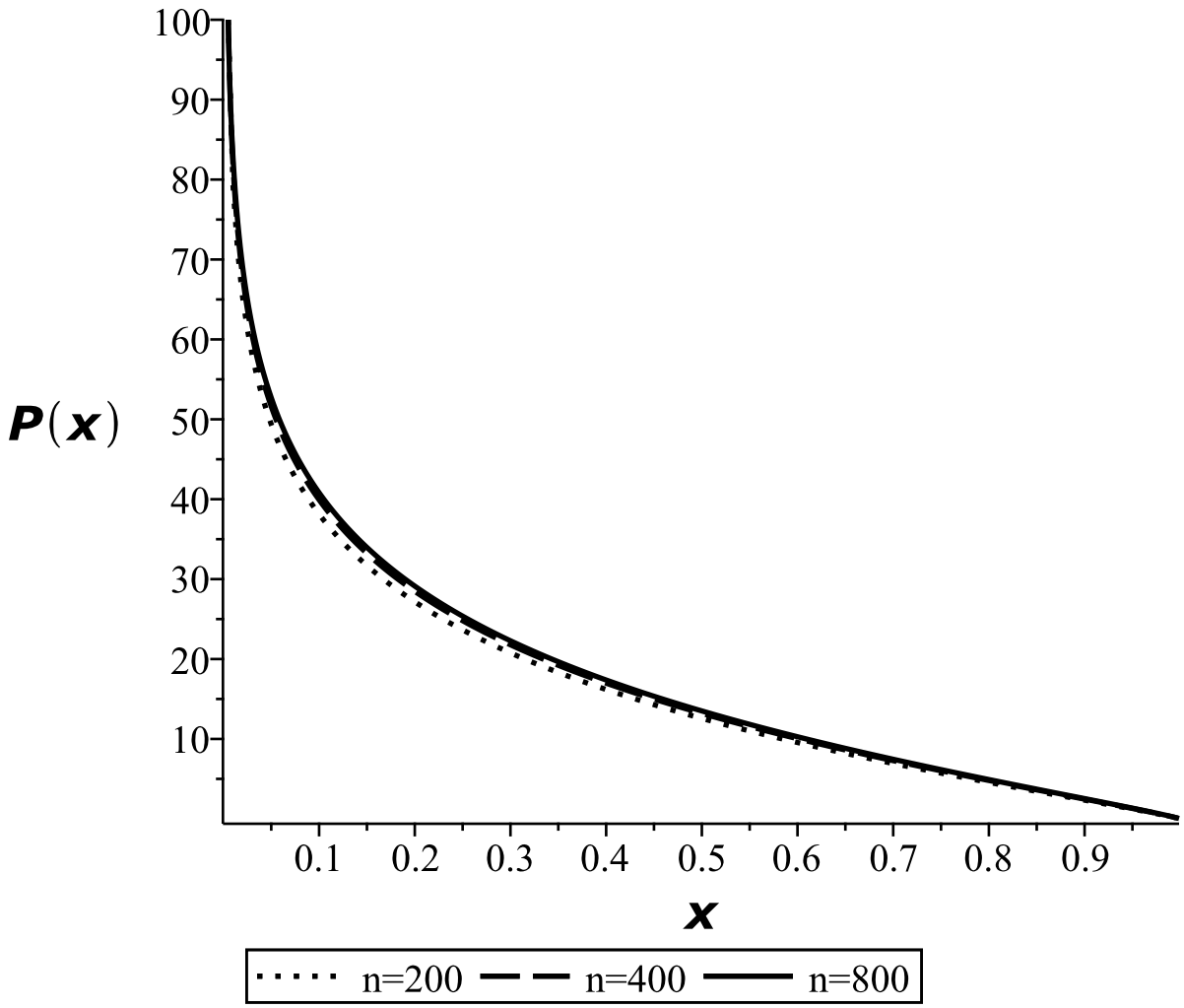
Error, (in plot) expecting 1 legend but got 3
Error, (in plot) expecting 1 legend but got 3



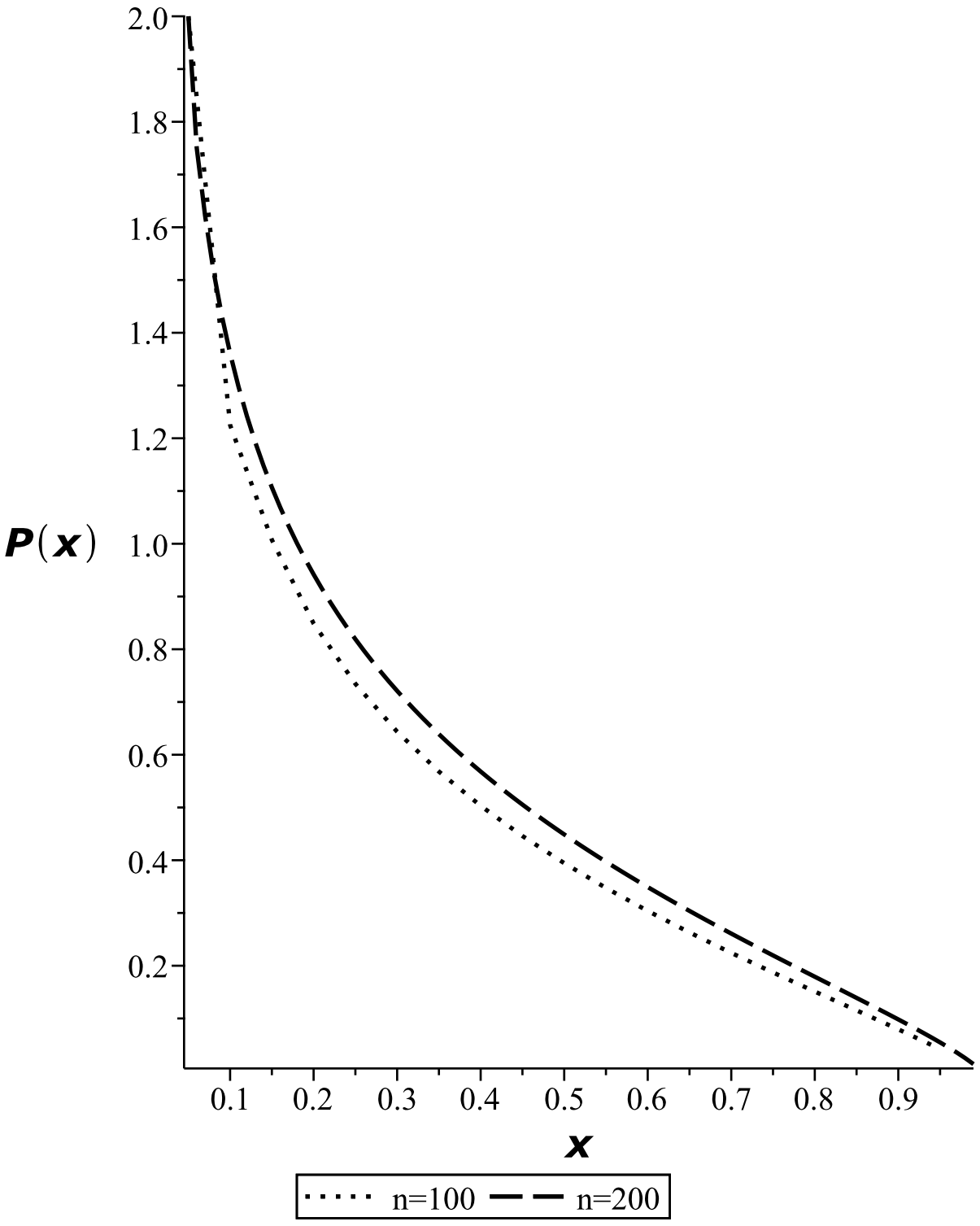

```
> SolN100[1,2]
```

100 **(2)**

```
>
  NumerPlot:=plot([SolN100,SolN200,SolN400],linestyle = [dot, dash,
  solid],thickness=2,color=black,labels=['x', P('x')],labelfont =
  ["Verdana",bold, 14],legend=["n=200","n=400","n=800"],legendstyle
  = [font = ["Verdana",bold, 12]])
```




```
> Nx100and200:=plot([SolN100,SolN200],linestyle = [dot, dash],
  thickness=2,color=black,labels=['x', P('x')],labelfont =
  ["Verdana",bold, 14],legend=["n=100","n=200"],legendstyle = [font
  = ["Verdana",bold, 12]]):
  display(Nx100and200);
```

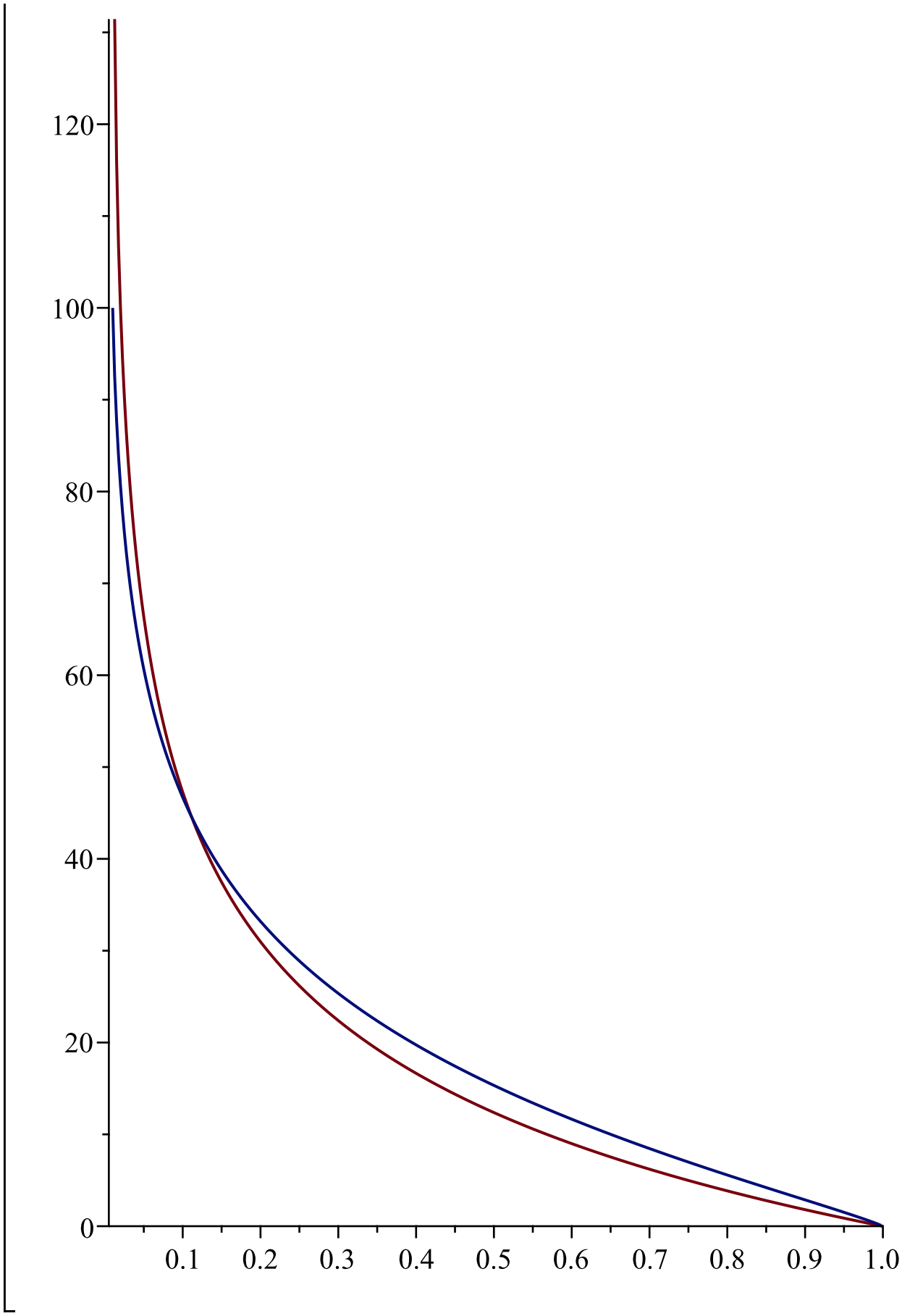



```
> SolN100:=[Solver(200,0.8,1)]:
```

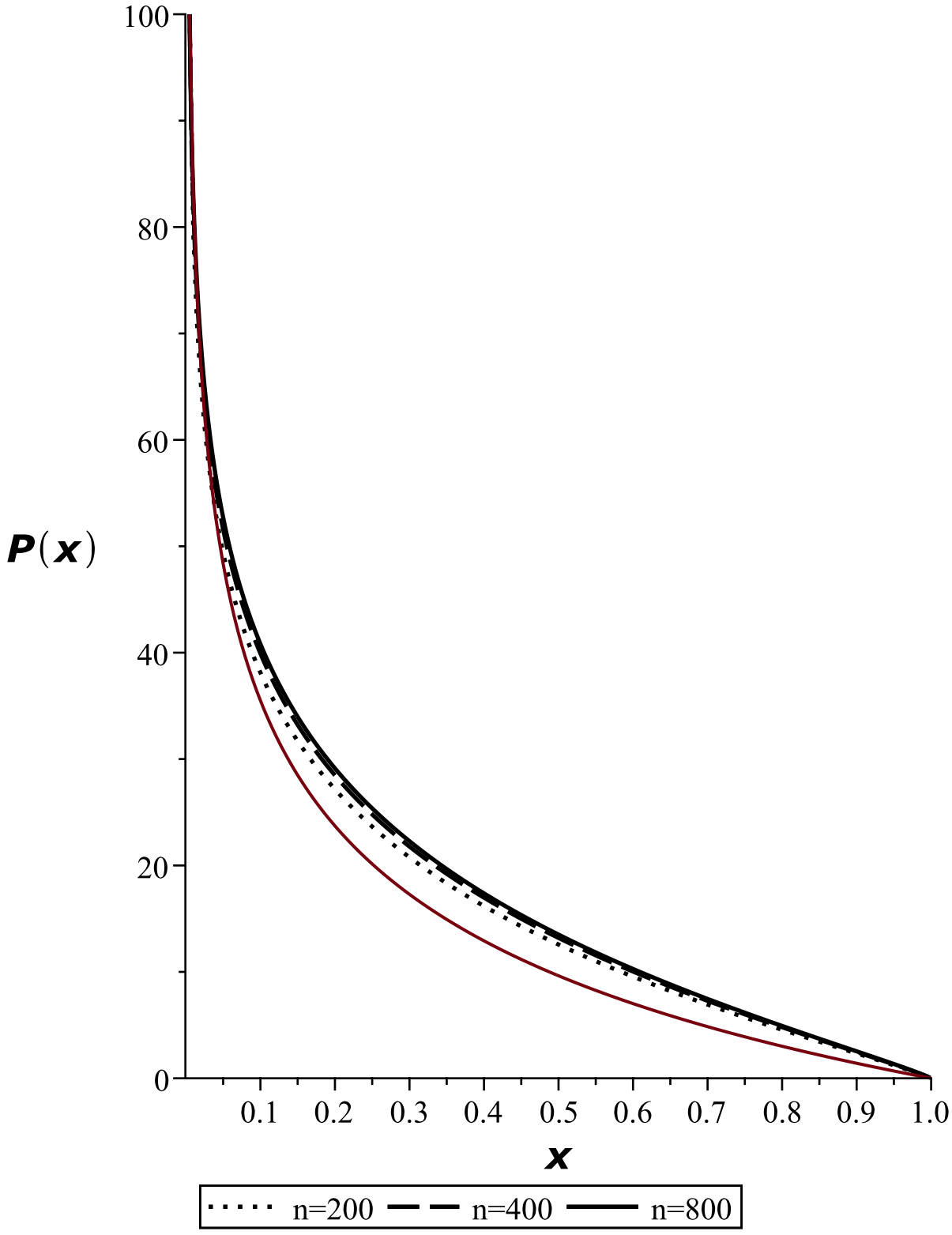
0.009975000000 **(3)**

```
> ExactSolver
> ExactSolver:= proc(nX,K,n1)
  global r0,R,alpha,Pn1,Pright:
  local RightX, LeftX, shagX, X, B, ExactSol, C1, C2,
  ExactSolveDef, p:
```

```
  local i,n,k:

  #
  RightX:=R:
  LeftX:=r0:
  X[0]:=LeftX:
  shagX:=evalf((RightX-LeftX)/nX):
  for i from 1 to nX do
            X[i] := evalf(X[i-1]+shagX):
  end do:


  #
  #ExactSol:=c2-c1*evalf(int(GAMMA(1/2)*(1/2-alpha/2)/GAMMA(3/2-
  alpha/2)/GAMMA(1-alpha/2)/R^((1+alpha)/2)*hypergeom([1, 1/2], [1-
  alpha/2], 1-x/R)*powFrac((x/R-1),(-1/4))/(2*sqrt(x)),x=r0..y));

  B[alpha]:=alpha^2*GAMMA(1+alpha/2)*GAMMA(1/2-alpha/2)/GAMMA(1/2+
  alpha/2);
  ExactSol:=c1/y^(alpha/2)*(1+B[alpha]*(X[n1]/y)^((alpha+1)/2)*
  hypergeom([1/2,1+alpha/2,1+alpha/2], [1,3/2], X[n1]/y))+c2;

  #
  solve({Pn1=evalf(eval(ExactSol,y=X[n1]+X[n1])),Pright=evalf(eval
  (ExactSol,y=R))});
  C1:=subs(%[1],c1): C2:=subs(%%[2],c2):  subs(c1=C1,c2=C2,
  ExactSol);
  ExactSolveDef:=subs(c1=C1,c2=C2,ExactSol):

  for i from 1 to nX do
            p[i] := evalf(eval(ExactSolveDef,y=X[i])):
  end do:

  #exactPlot:=plot(ExactSolveDef,y=SolN100[1,1]..R,color=black,
  l e g e n d = " "        ,legendstyle = [font = ["Verdana",bold, 12]
  ],thickness=3);


  seq([X[i],evalc(Re(p[i]))],i=n1..nX);
  end proc:
> AnalytPlot:=plot([[ExactSolver(400,0.8,4)],[Solver(400,0.8,4)]]);
                              0.01000000990
```


120
100
80
60
40
20
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0

```
> ExactSolver(100,0.8,1)
> display(NumerPlot,AnalytPlot);
```




```
>
```