```
> restart;
  read("c:/appl/appl7.txt");
```

PROCEDURES:

AllPermutations(n), AllCombinations(n, k), Benford(X), BootstrapRV(Data),
CDF:CHF:HF:IDF:PDF:SF(X, [x])), CoefOfVar(X), Convolution(X, Y),
ConvolutionIID(X, n), CriticalPoint(X, prob), Determinant(MATRIX), Difference(X, Y),
Display(X), ExpectedValue(X, [g]), KSTest(X, Data, Parameters), Kurtosis(X),
Maximum(X, Y), MaximumIID(X, n), Mean(X), MGF(X), Minimum(X, Y),
MinimumIID(X, n), Mixture(MixParameters, MixRVs),
MLE(X, Data, Parameters, [Rightcensor]), MLENHPP(X, Data, Parameters, obstime),
MLEWeibull(Data, [Rightcensor]), MOM(X, Data, Parameters),
NextCombination(Previous, size), NextPermutation(Previous), OrderStat(X, n, r, ["wo"]),
PlotDist(X, [low], [high]), PlotEmpCDF(Data, [low], [high]),
PlotEmpCIF(Data, [low], [high]), PlotEmpSF(Data, Censor),
PlotEmpVsFittedCDF(X, Data, Parameters, [low], [high]),
PlotEmpVsFittedCDF(X, Data, Parameters, [low], [high]),
PlotEmpVsFittedSF(X, Data, Parameters, Censor, low, high),
PPPlot(X, Data, Parameters), Product(X, Y), ProductIID(X, n),
QQPlot(X, Data, Parameters), RangeStat(X, n, ["wo"]), Skewness(X), Transform(X, g),
Truncate(X, low, high), Variance(X), VerifyPDF(X)

Procedure Notation:
X and Y are random variables
Greek letters are numeric or symbolic parameters
x is numeric or symbolic
n and r are positive integers, n >= r
low and high are numeric
g is a function
Brackets [] denote optional parameters
"double quotes" denote character strings
MATRIX is a 2 x 2 array of random variables
A capitalized parameter indicates that it must be
entered as a list --> ex. Data := [1, 12.4, 34, 52.45, 63]

Variate Generation:

ArcTanVariate(alpha, phi), BinomialVariate(n, p, m), ExponentialVariate(lambda),
NormalVariate(mu, sigma), UniformVariate(), WeibullVariate(lambda, kappa, m)

DATA SETS:

BallBearing, HorseKickFatalities, Hurricane, MP6, RatControl, RatTreatment, USSHalfBeak

ArcSinRV(), ArcTanRV(alpha, phi), BetaRV(alpha, beta), CauchyRV(a, alpha), ChiRV(n),

ChiSquareRV(n), ErlangRV(lambda, n), ErrorRV(mu, alpha, d), ExponentialRV(lambda), ExponentialPowerRV(lambda, kappa), ExtremeValueRV(alpha, beta), FRV(n1, n2), GammaRV(lambda, kappa), GeneralizedParetoRV(gamma, delta, kappa), GompertzRV(delta, kappa), HyperbolicSecantRV(), HyperExponentialRV(p, l), HypoExponentialRV(l), IDBRV(gamma, delta, kappa), InverseGaussianRV(lambda, mu), InvertedGammaRV(alpha, beta), KSRV(n), LaPlaceRV(omega, theta), LogGammaRV(alpha, beta), LogisticRV(kappa, lambda), LogLogisticRV(lambda, kappa), LogNormalRV(mu, sigma), LomaxRV(kappa, lambda), MakehamRV(gamma, delta, kappa), MuthRV(kappa), NormalRV(mu, sigma), ParetoRV(lambda, kappa), RayleighRV(lambda), StandardCauchyRV(), StandardNormalRV(), StandardTriangularRV(m), StandardUniformRV(), TRV(n), TriangularRV(a, m, b), UniformRV(a, b), WeibullRV(lambda, kappa)

Error, attempting to assign to `DataSets` which is protected.
Try declaring `local DataSets`; see ?protect for details.

```
> bf := RayleighRV(1);
  bfname := "RayleighRV(1)";
```

$$bf := \left[\left[x \to 2\,x\,e^{-x^2}\right], [0, \infty], ["\text{Continuous}", "\text{PDF}"]\right]$$

$$bfname := \text{RayleighRV(1)} \quad \textbf{(1)}$$

```
> #plot(1/csch(t)+1, t = 0..0.0010);
  #plot(diff(1/csch(t),t), t=0..0.0010);
  #limit(1/csch(t), t=0);
> solve(exp(-t) = y, t);
```

$$-\ln(y) \quad \textbf{(2)}$$

```
> # discarded  -ln(t + 1),  t-> csch(t),t->arccsch(t),t -> tan(t),
> glist := [t -> t^2 , t -> sqrt(t),  t -> 1/t,  t -> arctan(t),  t
  -> exp(t), t -> ln(t), t -> exp(-t), t -> -ln(t),  t -> ln(t+1),
  t -> 1/(ln(t+2)), t -> tanh(t), t -> sinh(t), t -> arcsinh(t),
  t-> csch(t+1),t->arccsch(t+1), t-> 1/tanh(t+1), t->  1/sinh(t+1),
   t->  1/arcsinh(t+1), t->  1/csch(t)+1, t-> tanh(1/t), t->csch
  (1/t), t-> arccsch(1/t), t-> arctanh(1/t)  ]:
  base := t -> PDF(bf, t):
  print(base(x)):

  for i from 22 to 22(glist) do
     print( "i is", i, "  ----------------------------------------
  ----------------------------------------------------------------
  -------------");
     g := glist[i]:
     l := bf[2][1];
     u := bf[2][2];
     Temp := Transform(bf, [[unapply(g(x), x)],[l,u]]);


    #print( "l and u", l, u );
    #print("g(x)", g(x), "base", base(x),bfname);
```

```
    print("f(x)", PDF(Temp, x));
    #print("F(x)", CDF(Temp, x));
    #print("IDF(x)", IDF(Temp));
    #print("S(x)", SF(Temp, x));
    print("h(x)", HF(Temp, x));
    #print("mean and variance", Mean(Temp), Variance(Temp));
    #assume(r > 0); mf := int(x^r*PDF(Temp, x), x = Temp[2][1] ..
  Temp[2][2]);
    #print("MF", mf);
    #print("MGF", MGF(Temp));
    PlotDist(PDF(Temp), 0, 40);
    PlotDist(HF(Temp), 0, 40);
    latex(PDF(Temp,x));
    #print("transforming with", [[x->g(x)],[0,infinity]]);
    #X2 := Transform(bf, [[x->g(x)],[0,infinity]]);
    #print("pdf of X2 = ", PDF(X2,x));
    #print("pdf of Temp = ", PDF(Temp,x));
  od;
```

$$2\,x\,e^{-x^2}$$

"i is", 22,

" ----------"

$$g := t \to \operatorname{arccsch}\left(\frac{1}{t}\right)$$

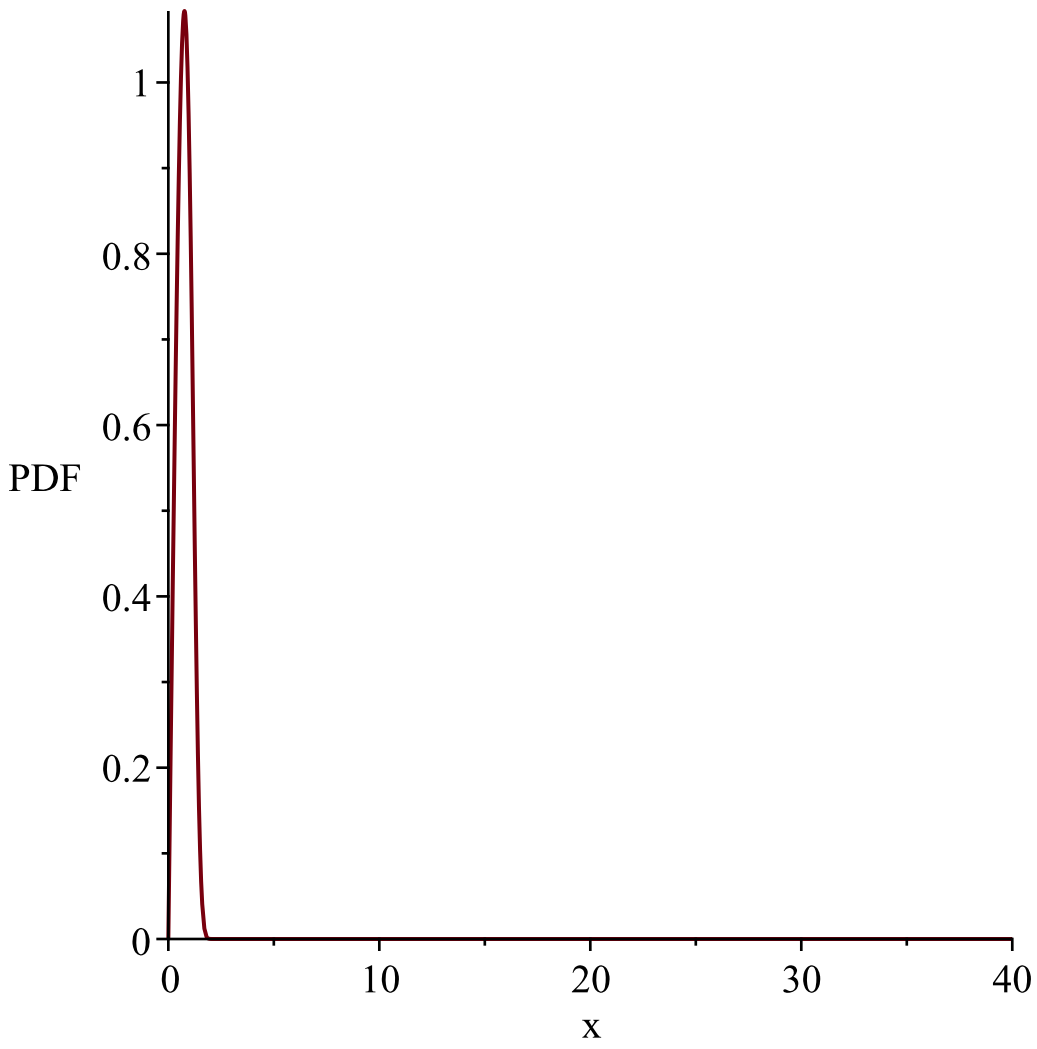
$$l := 0$$

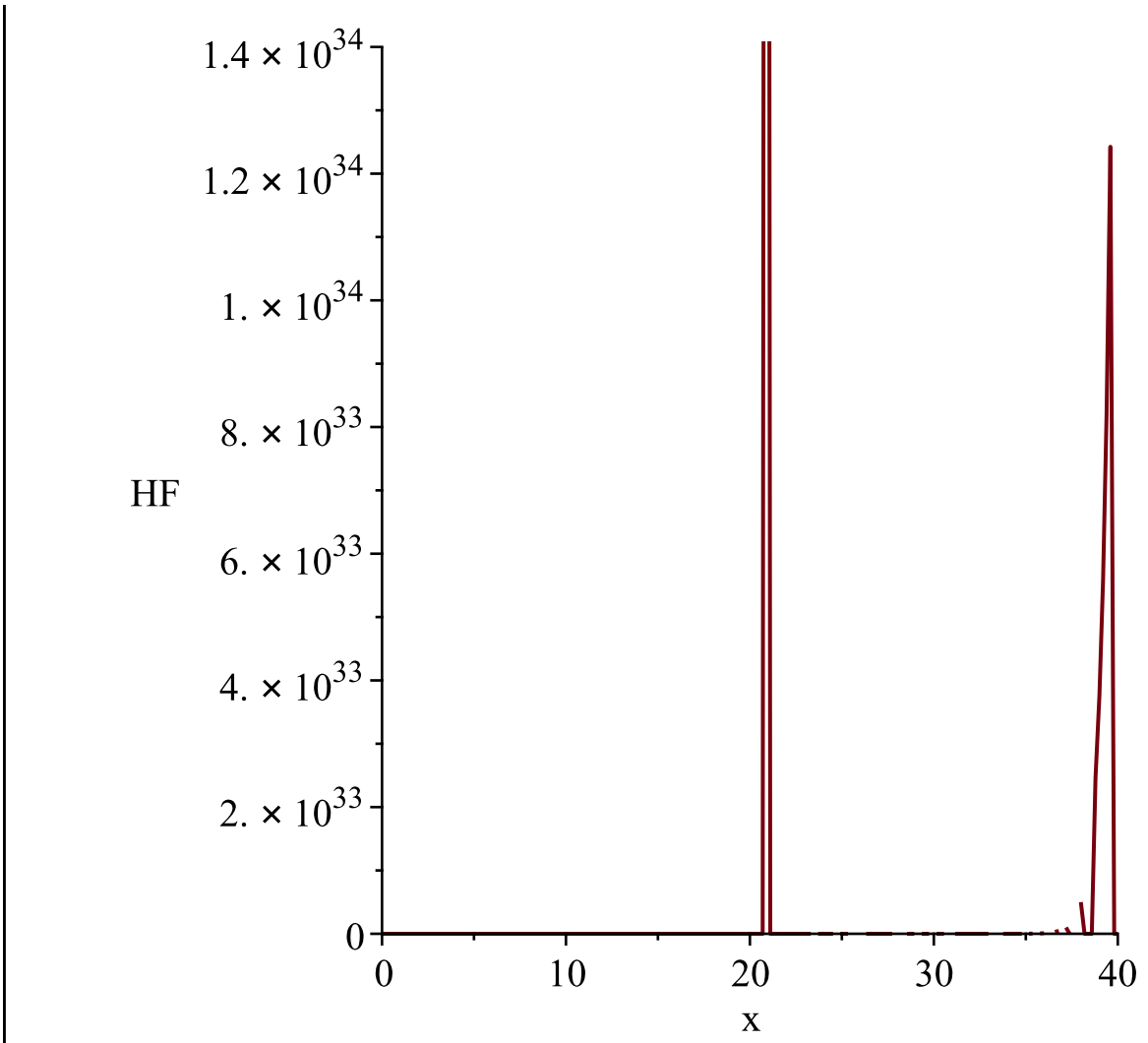
$$u := \infty$$

$$Temp := \left[\left[y\sim \to 2\,e^{-\sinh(y\sim)^2}\cosh(y\sim)\,\sinh(y\sim)\right], [0, \infty], ["Continuous", "PDF"]\right]$$

"f(x)", $2\,e^{-\sinh(x)^2}\cosh(x)\,\sinh(x)$

"h(x)", $2\,e^{-\cosh(x)^2+\frac{1}{2}+\frac{1}{4}e^{2x}+\frac{1}{4}e^{-2x}}\cosh(x)\,\sinh(x)$


1
0.8
0.6
PDF
0.4
0.2
0
0
10
20
30
40
x




```
2\,{{\rm e}^{- \left( \sinh \left( x \right)  \right) ^{2}}}
\cosh
 \left( x \right) \sinh \left( x \right)
```