

Stable Distributions in Mathematica

Robert H. Rimmer
John P. Nolan

A *Mathematica* package for the computation of stable distributions is demonstrated. Calculations of stable density, distribution, and quantile functions can be performed and stable random variables generated. Data can be fit to stable parameters by the maximum likelihood method. Extreme precision calculations are possible. The *StableDistribution* package and documentation are provided in the Additional Material section.

■ Introduction

Stable distributions are a rich class of probability distributions that allow skewness and heavy tails and have many intriguing mathematical properties. The class was characterized by Paul Levy in his study of sums of independent, identically distributed terms in the 1920s [1]. Stable distributions have been proposed as models for many types of physical and economic systems. There are solid theoretical reasons for expecting a non-Gaussian stable model, for example, reflection off a rotating mirror yielding a Cauchy distribution, hitting times for a Brownian motion yielding a Levy distribution, and the gravitational field of stars yielding the Holtsmark distribution. The generalized central limit theorem states that the only possible nontrivial limit of normalized sums of independent, identically distributed terms is stable. It is argued that some observed quantities are the sum of many small terms; hence, a stable model should be used to describe such systems. For a detailed exposition of stable distributions and their applications, see [2].

The lack of formulae for densities and distribution functions for the full parameter range has limited the use of stable distributions by practitioners. Recently high-speed processors have made numerical calculations of stable functions feasible, and interest in their use has been increasing, especially in the study of financial markets based on Mandelbrot's work [3, 4, 5]. Rose and Smith [6] have used *Mathematica* to generate special functions to calculate stable densities and cumulative distributions, but these special functions exist only for certain rational values of stable parameters. Our program takes advantage of the Zolotarev integral representation of the stable density and the capabilities of the function, `NIntegrate`, to accurately calculate stable distributions for nearly the whole

parameter range. Its availability in *Mathematica* should make stable distributions accessible to a wide audience across all commonly used computing environments.

■ Parameterizations of Stable Laws

A general stable distribution requires four parameters to describe it:

- index of stability or characteristic exponent $\alpha \in (0, 2]$
- skewness parameter $\beta \in [-1, 1]$
- scale parameter $\gamma > 0$
- location parameter $\delta \in \mathbb{R}$

There are multiple parameterizations for stable laws, and much confusion has been caused by these different parameterizations. This package includes two parameterizations. The characteristic functions in the $S(\alpha, \beta, \gamma, \delta; 0)$ and $S(\alpha, \beta, \gamma, \delta; 1)$ parameterizations follow.

$S(\alpha, \beta, \gamma, \delta; 0)$:

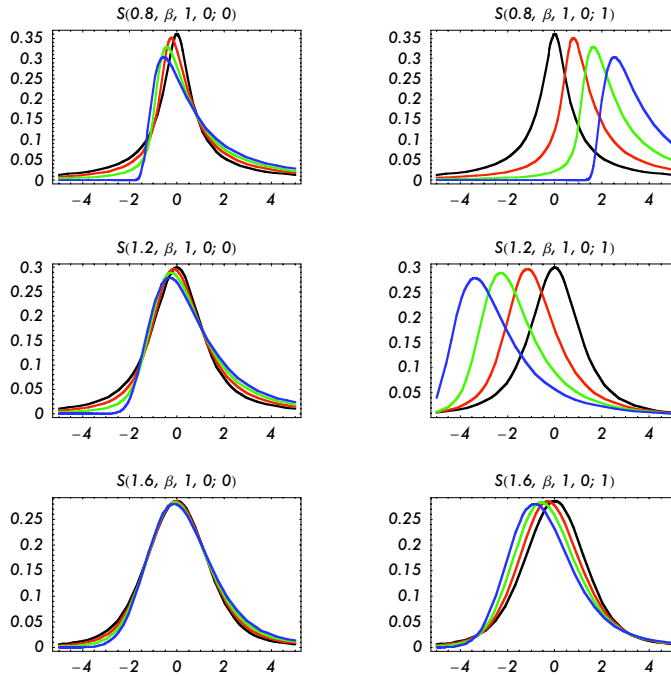
$$\begin{aligned} \exp\left(i u \delta - \gamma^\alpha \text{Abs}[u]^\alpha \left(1 + i \beta (-1 + \text{Abs}[u \gamma]^{1-\alpha}) \text{Sign}[u] \text{Tan}\left[\frac{\pi \alpha}{2}\right]\right)\right) & \quad \alpha \neq 1 \\ \exp\left(i u \delta - \frac{\gamma \text{Abs}[u] (\pi + 2 i \beta \text{Log}[\text{Abs}[u \gamma]] \text{Sign}[u])}{\pi}\right) & \quad \alpha = 1 \end{aligned}$$

and

$S(\alpha, \beta, \gamma, \delta; 1)$:

$$\begin{aligned} \exp\left(i u \delta - \gamma^\alpha \text{Abs}[u]^\alpha \left(1 - i \beta \text{Sign}[u] \text{Tan}\left[\frac{\pi \alpha}{2}\right]\right)\right) & \quad \alpha \neq 1 \\ \exp\left(i u \delta - \gamma \text{Abs}[u] \left(1 + \frac{2 i \beta \text{Log}[\text{Abs}[u]] \text{Sign}[u]}{\pi}\right)\right) & \quad \alpha = 1 \end{aligned}$$

The $S(\alpha, \beta, \gamma, \delta; 0)$ parameterization is the best choice for numerical computation of stable distributions, since it has the simplest form for the characteristic function that is continuous in all four parameters. The $S(\alpha, \beta, \gamma, \delta; 1)$ parameterization has been commonly used in economic literature and has the property that the location parameter, δ , is the mean when $\alpha > 1$. The following graphics array demonstrates these two parameterizations. The scale parameter, γ , has been set to 1 and the location parameter, δ , is 0. Each graph shows varying values of β (0, 1/3, 2/3, 1 in black, red, green, and blue colors, respectively) at the value of α and parameterization type listed on the graph. Negative values of β would give mirror image graphs. When α is near 1, the mode in the $S(\alpha, \beta, \gamma, \delta; 1)$ parameterization moves dramatically; the $S(\alpha, \beta, \gamma, \delta; 0)$ parameterization progressively reduces this motion as α approaches 1.



■ Using the Package

The *StableDistribution* package provides several basic functions needed for the calculation of stable distributions. They are all designed with identical input, output, and default formats. The package comes with a Help Browser that explains in more detail how each function works. It also includes a few more functions that are not described in this paper. The package allows both the 0 and 1 parameterizations for all the following functions, but in the examples and discussion only $S(\alpha, \beta, \gamma, \delta; 1)$ will be used.

Place the unzipped folder, *StableDistribution* (see Additional Material), in the *Mathematica* applications folder. After rebuilding the Help Browser index, documentation will be available in the Help Browser Add-ons & Links category. The following command loads the program.

```
In[1]:= << StableDistribution`SMath`
```

`SPDF[x, {α, β, γ, δ}, param]` calculates density of a stable $S(\alpha, \beta, \gamma, \delta; \text{param})$ distribution at the value in x .

`SCDF[x, {α, β, γ, δ}, param]` calculates cumulative distribution function of a stable $S(\alpha, \beta, \gamma, \delta; \text{param})$ distribution at the value in x .

`SQuantile[p, { α , β , γ , δ }, param]` calculates the value of x for a stable $S(\alpha, \beta, \gamma, \delta; \text{param})$ distribution for a quantile p .

`SRandom[n, { α , β , γ , δ }, param]` generates n stable $S(\alpha, \beta, \gamma, \delta; \text{param})$ random variables.

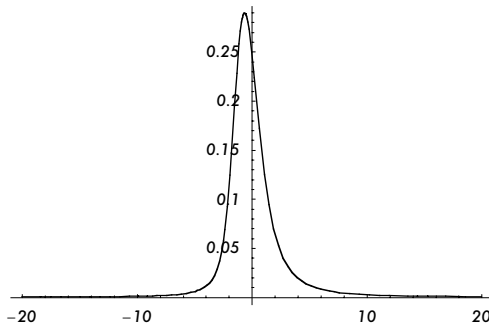
`fMLFit[x_List, param]` fits a $S(\alpha, \beta, \gamma, \delta; \text{param})$ stable distribution to the data in x , using a maximum likelihood method.

`D1SPDF[x, { α , β , γ , δ }, param]` calculates the first derivative of the density of a stable $S(\alpha, \beta, \gamma, \delta; \text{param})$ distribution at the value in x .

`SMode[{ α , β , γ , δ }, param]` returns the location of the mode of a $S(\alpha, \beta, \gamma, \delta; \text{param})$ stable distribution.

The SPDF function calculates the stable density using a variation of the Zolotarev integral transformation [7]. This transformation permits accurate calculations of both the density and distribution functions across nearly the whole domain of stable parameters. Attempts to simply numerically integrate the inverse Fourier transform of the characteristic function run into difficulties when $\alpha < 1$ or when x is large.

```
In[2]:= Plot[SPDF[x, {1.3, 0.5, 1, 0.5}], {x, -20, 20}];
```



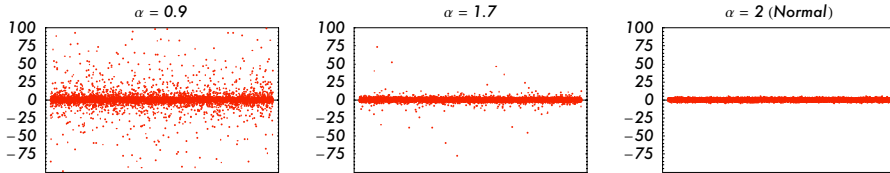
Chambers, Mallows, and Stuck [8] published an algorithm for the generation of stable random variables. It is implemented in this package as the function, `SRandom`. The following example generates a stable random sequence of 10 numbers where $p1$ is a parameter set, $\{\alpha, \beta, \gamma, \delta\}$.

```
In[3]:= p1 = {1.3, 0.5, 1, 0.5};
SRandom[10, p1]
```

```
Out[4]:= {-0.832182, -0.292768, 0.0465778, 9.15896, -10.9316,
0.33523, -0.345039, 0.104213, 0.140818, 3.89919}
```

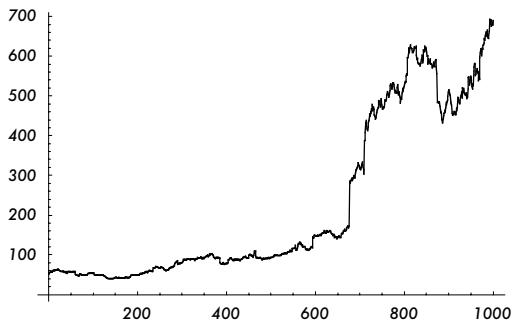
The `SRandom` generator makes it easy to experience the behavior of stable distributions. The following graphics array illustrates 10,000 pseudorandom

drawings from symmetric stable distributions with α set at 0.9, 1.7, and 2.0. Note how extreme outliers may be for lower values of α . The graphs are all at the same scale to illustrate the differences in dispersion, but at this scale some of the extreme points in the case of $\alpha = 0.9$ are not shown. A stable sample with $\alpha = 2$ has the same dispersion as the normal distribution. $\alpha = 1.7$ is typical of stock price returns.



The following example simulates what a series of daily stock market prices might look like if generated by a stable return model with a starting price of 50 and an expected annual return of 7% (there are about 252 trading days in a year); notice the sudden large jumps.

```
In[5]:= ListPlot[50 Exp[FoldList[Plus, 0, SRandom[1000,
    {1.7, -0.1, 0.012, Log[1.07] / 252}]]], PlotJoined -> True];
```



The *StableDistribution* package also has data analysis capability and can perform a fit to data. The `fMLFit` function provides a maximum likelihood fit to the data. The algorithm uses a fast Fourier transform of the characteristic function [9] to approximate the stable probability density function rapidly.

```
In[6]:= p1 = {1.4, 0.9, 1, -0.5}
s4 = SRandom[10000, p1];
p2 = fMLFit[s4]
```

```
Out[6]:= {1.4, 0.9, 1, -0.5}
```

```
FindMaximum::lstol :
```

```
The line search decreased the step size to within tolerance specified by
AccuracyGoal and PrecisionGoal but was unable to find a sufficient
increase in the function. You may need more than MachinePrecision
digits of working precision to meet these tolerances. More . . .
```

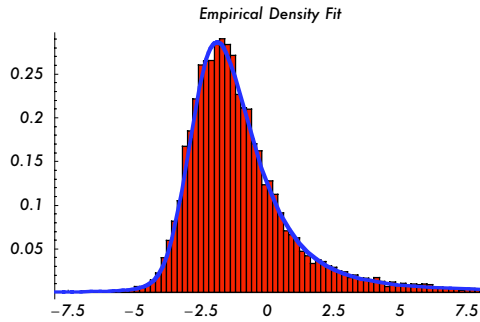
```
Out[8]:= {1.35908, 0.851204, 0.982561, -0.383498}
```

The following graph compares the fit to the empirical density of the sample.

```

In[9]:= << Graphics `Graphics`
empircdens = Histogram[s4, HistogramScale → 1,
  HistogramRange → {-8, 8}, DisplayFunction → Identity];
calcdens = Plot[SPDF[x, p2], {x, -8, 8}, PlotStyle →
  {Hue[0.67], Thickness[0.008]}, DisplayFunction → Identity];
Show[empircdens, calcdens, PlotLabel → "Empirical Density Fit",
  DisplayFunction → $DisplayFunction];

```

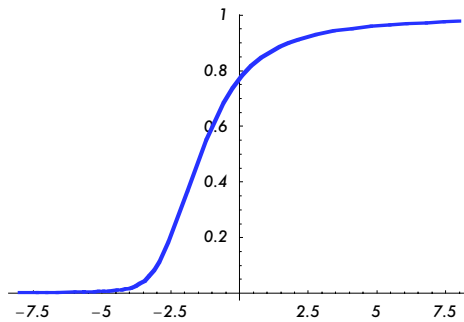


The plot of the distribution function follows.

```

In[13]:= Plot[SCDF[x, p2], {x, -8, 8},
  PlotStyle → {Hue[0.67], Thickness[0.008]};

```



The `SQuantile` function uses `FindRoot` to solve for x at a value of the SCDF. It can be used to print a table of cumulative probabilities. For the parameters in the preceding example:

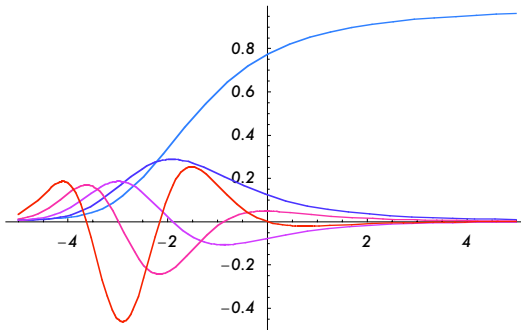
```
In[14]:= Table[{p, SQuantile[p, p1]}, {p, 0, 1, 0.1}] // TableForm
```

```
Out[14]//TableForm=
```

0	$-\infty$
0.1	-3.04994
0.2	-2.55465
0.3	-2.16935
0.4	-1.81086
0.5	-1.44195
0.6	-1.02835
0.7	-0.516218
0.8	0.221794
0.9	1.69672
1.	∞

The density and distribution functions use `NIntegrate` to make the calculations. A direct integral form for the first derivative of the density function is also provided, but further derivatives can also be calculated using the derivative function in *Mathematica*.

```
In[15]:= st1 = Table[Hue[i], {i, 0.6, 1, 0.1}];
f2[x_, a_] := D[SPDF[x, a], {x, 2}];
f3[x_, a_] := D[SPDF[x, a], {x, 3}];
Plot[Evaluate[
  {SCDF[x, p2], SPDF[x, p2], D1SPDF[x, p2], f2[x, p2], f3[x, p2]}],
{x, -5, 5}, PlotStyle -> st1, PlotRange -> All];
```



While the package is able to calculate most all of the range of stable distributions, it can at times be slow and may require adjustment of the input parameters to `NIntegrate` to obtain accurate results. With such adjustments, however, it is possible to make extremely precise calculations. The following example shows high-precision density calculations for values on the light tail of a maximally skewed stable distribution.

```
In[19]:= Table[{2^i, SPDF[2^i, {13/10, -1}, GaussPoints -> 30,
MaxRecursion -> 10, WorkingPrecision -> 64]}, {i, 6}] // TableForm
```

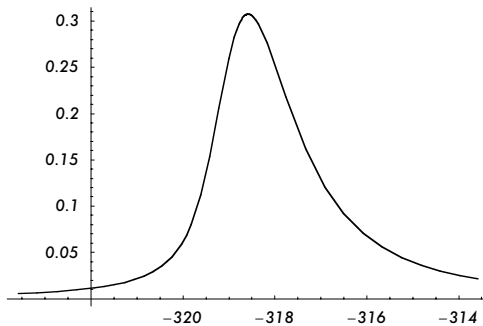
```
Out[19]//TableForm=
```

2	0.273549462258796189509775341539313131858185752986508966233
4	0.038598879516744249569924583917870716560367065955283947236
8	3.391979139387401921303888873307117869238901361946908063 $\times 10^{-25}$
16	1.292128069876843683121684007751601752432115597266917711 $\times 10^{-496}$
32	1.4907823276439596142503265579049423920572129735541813671 $\times 10^{-1006}$
64	1.28876881056847406907355147746013211220193021391578303 $\times 10^{-201721}$

The `SMode` function can be used to locate the mode of a stable distribution; this is particularly useful when α is close to 1, where the mode may be quite distant from the origin. The function uses `FindRoot` to determine where the first derivative of the stable density is equal to 0.

```
In[20]:= p3 = {1.001, 0.5};
md = SMode[p3]
Plot[SPDF[x, p3], {x, md - 5, md + 5}];
```

```
Out[21]= -318.575
```



■ Program Accuracy, Limitations, and Versatility

The functions illustrated here are designed to have the look and feel of true mathematical functions. `SPDF` uses the Zolotarev transformation of the inverse Fourier transform of the characteristic function in the $S(\alpha, \beta, \gamma, \delta; 0)$ parameterization. The integrand is represented in the program symbolically so that transformation to the $S(\alpha, \beta, \gamma, \delta; 1)$ parameterization is exact. The `SCDF` and `D1SPDF` integrands are created by taking the integral and derivative of the `SPDF` integrand with respect to x before proceeding with transform integration. The function, `NIntegrate`, gives a numerical approximation to the integral. It is important to remember that `NIntegrate` may not always come up with the correct answer. This is particularly true if the integrand has a sharp narrow spike. The integrand for the stable density can be problematic. The integrand is continuous and does not oscillate over the interval of integration; it begins with a value of 0, rises to a peak, $\frac{1}{e}$, and falls to 0 at the other end of the interval. Unfortunately the peak for

some values of x , α , and β can be quite narrow and it is possible for `NIntegrate` to miss the peak with the default settings, resulting in an inaccurate approximation. Since the integrand is otherwise well behaved, it appears to be almost always possible to overcome this limitation and produce calculations to extreme accuracy. When the warning messages for `NIntegrate` are turned on, the first message will generally give a hint as to how to proceed. If the goal is high precision, the first strategy should be to increase the `WorkingPrecision` option. If machine precision is acceptable but the problem is that the peak is missed (most common with probability density function calculations), the best strategy is to increase the `GaussPoints` option. Even though the integrand does not contain a singularity, sometimes it will be necessary to increase the `SingularityDepth` option to catch the peak.

```
In[23]:= OnNIntMsg ; (*Turn on suppressed NIntegrate warning messages*)
```

```
In[24]:= SPDF[0.001, {1.3, 0}]
```

```
NIntegrate::ncvb :
NIntegrate failed to converge to prescribed accuracy after 7
recursive bisections in SMath`Private`θ near
SMath`Private`θ = 0.0006014037137896312`. More . . .
```

```
Out[24]= 0.293983
```

Here we could increase the recursion limit as suggested by the error message, but for these functions it is often a better idea to start with `GaussPoints` which sets a higher number of initial points, thus requiring less recursive subdivision. The difference between the calculations is small.

```
In[25]:= SPDF[0.001, {1.3, 0}, GaussPoints -> 15]
% - %
```

```
Out[25]= 0.293983
```

```
Out[26]= -4.14103 × 10-10
```

For extreme precision, start with working precision. As the output precision will be limited by the specification of the precision of the input values, it is simplest to use rational numbers. `GaussPoints` and/or `MaxRecursion` will also likely need to be increased. The default setting for `PrecisionGoal` in `NIntegrate` is `WorkingPrecision` minus 10 digits.

```
In[27]:= Table[SPDF[1/1000, {13/10, 0}, GaussPoints → 30, MaxRecursion → 10,
WorkingPrecision → p], {p, 16, 64, 4}] // TableForm
```

```
Out[27]//TableForm=
0.2939835
0.2939834576205198
0.2939834576205198
0.2939834576205197617402
0.29398345762051976174019
0.293983457620519761740188001
0.29398345762051976174018800081232
0.29398345762051976174018800081231973
0.29398345762051976174018800081231973391
0.29398345762051976174018800081231973390709049
0.29398345762051976174018800081231973390709048777
0.293983457620519761740188000812319733907090487774355
0.2939834576205197617401880008123197339070904877743551734
```

When the default `NIntegrate` parameters are used, the program may fail to give accurate results when α is close to 1 or when α is 1 and β is close to 0. These limitations can often be overcome by changing the optional arguments to `NIntegrate`. With the warning messages for `NIntegrate` turned on, *Mathematica* will give notification that requested precision has not been achieved.

```
In[28]:= SPDF[-10, {1.01, 0.5, 1, 0.5}]
```

```
NIntegrate::ploss :
Numerical integration stopping due to loss of precision. Achieved neither
the requested PrecisionGoal nor AccuracyGoal; suspect one of the
following: highly oscillatory integrand or the true value of the integral
is 0. If your integrand is oscillatory on a (semi-)infinite interval
try using the option Method->Oscillatory in NIntegrate. More . . .
```

```
Out[28]= 1.34339 × 10-27
```

```
In[29]:= SPDF[-10, {1.01, 0.5, 1, 0.5}, GaussPoints → 30]
```

```
Out[29]= 0.00108592
```

```
In[30]:= Plot[SPDF[x, {1.005, 0.02}], {x, -20, 20}, PlotRange → All];
```

```
NIntegrate::ploss :
Numerical integration stopping due to loss of precision. Achieved neither
the requested PrecisionGoal nor AccuracyGoal; suspect one of the
following: highly oscillatory integrand or the true value of the integral
is 0. If your integrand is oscillatory on a (semi-)infinite interval
try using the option Method->Oscillatory in NIntegrate. More . . .
```

```
NIntegrate::ploss :
Numerical integration stopping due to loss of precision. Achieved neither
the requested PrecisionGoal nor AccuracyGoal; suspect one of the
following: highly oscillatory integrand or the true value of the integral
is 0. If your integrand is oscillatory on a (semi-)infinite interval
try using the option Method->Oscillatory in NIntegrate. More . . .
```

`NIntegrate::ploss` :
 Numerical integration stopping due to loss of precision. Achieved neither the requested `PrecisionGoal` nor `AccuracyGoal`; suspect one of the following: highly oscillatory integrand or the true value of the integral is 0. If your integrand is oscillatory on a (semi-)infinite interval try using the option `Method->Oscillatory` in `NIntegrate`. [More . . .](#)

`General::stop` : Further output of `NIntegrate::ploss` will be suppressed during this calculation. [More . . .](#)

`NIntegrate::ncvb` :
`NIntegrate` failed to converge to prescribed accuracy after 7 recursive bisections in `SMath'Private'\theta` near `SMath'Private'\theta = 1.5092880787986058`. [More . . .](#)

`NIntegrate::slwcon` :
 Numerical integration converging too slowly; suspect one of the following: singularity, value of the integration being 0, oscillatory integrand, or insufficient `WorkingPrecision`. If your integrand is oscillatory try using the option `Method->Oscillatory` in `NIntegrate`. [More . . .](#)

`NIntegrate::ncvb` :
`NIntegrate` failed to converge to prescribed accuracy after 7 recursive bisections in `SMath'Private'\theta` near `SMath'Private'\theta = 1.5092880787986058`. [More . . .](#)

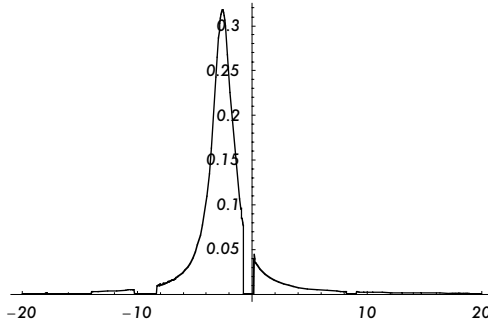
`NIntegrate::slwcon` :
 Numerical integration converging too slowly; suspect one of the following: singularity, value of the integration being 0, oscillatory integrand, or insufficient `WorkingPrecision`. If your integrand is oscillatory try using the option `Method->Oscillatory` in `NIntegrate`. [More . . .](#)

`NIntegrate::ncvb` :
`NIntegrate` failed to converge to prescribed accuracy after 7 recursive bisections in `SMath'Private'\theta` near `SMath'Private'\theta = 1.5092880787986058`. [More . . .](#)

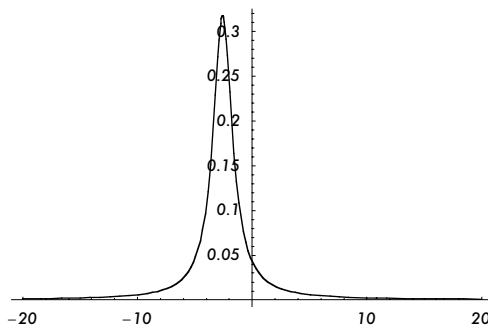
`General::stop` : Further output of `NIntegrate::ncvb` will be suppressed during this calculation. [More . . .](#)

`NIntegrate::slwcon` :
 Numerical integration converging too slowly; suspect one of the following: singularity, value of the integration being 0, oscillatory integrand, or insufficient `WorkingPrecision`. If your integrand is oscillatory try using the option `Method->Oscillatory` in `NIntegrate`. [More . . .](#)

`General::stop` : Further output of `NIntegrate::slwcon` will be suppressed during this calculation. [More . . .](#)



```
In[31]:= Plot[SPDF[x, {1.005, 0.02}], GaussPoints -> 30, MaxRecursion -> 10],
{x, -20, 20}, PlotRange -> All];
```



The package's Help Browser gives additional examples to help handle error messages generated by the other functions. It contains a section that demonstrates other methods of calculating stable distributions using *Mathematica* and includes the derivation of the Zolotarev integral. There is also an algorithm to set up an interpolation method for approximating the density function for maximum likelihood calculations.

■ **MathLink Implementation**

The direct use of *Mathematica* for data analysis can be slow. For in-depth analysis of data, *STABLE* is an excellent *MathLink* package that includes many more functions and parameterizations. The *MathLink* program consists of compiled Fortran routines that are passed to *Mathematica* with wrapper functions that are compatible with the functions in the *StableDistribution* package. It solves the problem of the shape of the Zolotarev integrand by dividing it into intervals. The peak is found numerically and a narrow interval on either side of the peak is handled with a more involved routine than the other parts. Its compiled code can thus use simpler and faster numerical integration routines for greater speed than possible with *Mathematica*'s very versatile but complicated `NIntegrate` function. It is also bulletproof, at the cost of not allowing calculations when $\text{Abs}[\alpha - 1] < 0.01$ but rounding to $\alpha = 1$, and not permitting calculations when

$\alpha < 0.1$. It is limited to double-precision calculations. The interface can perform maximum likelihood calculations many times faster than the *Mathematica* package described earlier, using a very fast interpolation method. These fits are usually a minimum of 50 times faster than possible with the package described in this paper. The *MathLink* interface has been used with *Mathematica*'s *J/Link* to download stock market data from internet databases and automatically analyze the time series of thousands of stocks. The *MathLink* interface is available (currently for Windows only) from www.robustanalysis.com. The package described in this paper is designed so that both programs can be run simultaneously turning *Mathematica* into a powerful, stable workstation.

■ References

- [1] P. Levy, *Calcul des Probabilites*, Paris: Gauthier-Villars, 1925.
- [2] J. P. Nolan, *Stable Distributions: Models for Heavy Tailed Data*, Boston: Birkhauser, 2005. Chapter 1 provides a good introduction and is available at the American University Stable website academic2.american.edu/~jpnolan/stable/stable.html.
- [3] B. B. Mandelbrot, "The Variation of Certain Speculative Prices," *Journal of Business*, **36**(4), 1963 pp. 394–419.
- [4] B. B. Mandelbrot, "Correction of an Error in "The Variation of Certain Speculative Prices" (1963)," *Journal of Business*, **45**(4), 1972 pp. 542–543.
- [5] B. B. Mandelbrot, *Fractals and Scaling in Finance: Discontinuity, Concentration, Risk*, New York: Springer-Verlag, 1997.
- [6] C. Rose and M. D. Smith, *Mathematical Statistics with Mathematica*, Springer Texts in Statistics, New York: Springer-Verlag, 2002.
- [7] J. P. Nolan, "Numerical Calculation of Stable Densities and Distribution Functions," *Communications in Statistics-Stochastic Models*, **13**(4), 1997 pp. 759–774.
- [8] J. M. Chambers, C. L. Mallows, and B. W. Stuck, "A Method for Simulating Stable Random Variables," *Journal of the American Statistical Association*, **71**(354), 1976 pp. 340–344.
- [9] S. T. Rachev and S. Mittnik, *Stable Paretian Models in Finance*, New York: John Wiley, 2000.
- [10] J. P. Nolan, "Stable MathLink Package," (Mar 18, 2005) www.robustanalysis.com.
- [11] V. M. Zolotarev, "One-Dimensional Stable Distributions," Translation of Mathematical Monographs, Vol. 65, American Mathematical Society, 1986. (Translation of the original 1983 Russian).

■ Additional Material

StableDistribution.zip

Available at www.mathematica-journal.com/issue/v9i4/download.

About the Authors

Robert H. Rimmer received his M.D. degree from the State University of New York. He is a clinical cardiologist with an interest in mathematics. He became interested in stable distributions after reading *The Fractal Geometry of Nature* by B. B. Mandelbrot in the 1980s. He has been programming with *Mathematica* as a hobby since 1988.

John P. Nolan received his Ph.D. in mathematics from the University of Virginia. He is a professor at American University in Washington, DC, where he teaches and does research on stable distributions and statistical genetics.

Robert H. Rimmer

*HC 4, Box 3-C
Payson, AZ 85541-9571
rrimmer@npgcable.com*

John P. Nolan

*Math/Stat Department
227 Gray Hall
American University
4400 Massachusetts Avenue, NW
Washington, DC 20016-8050
jpnolan@american.edu*