Using `tsallis`

The Tsallis distortion, a Beta mixture of TVaRs with a closed-form Gini index, and the extended calibration set

Stephen J. Mildenhall

2026-11-06

The **Tsallis distortion**[1] of order $\alpha > 0$ is the mixture of the TVaR distortions $\min(s/t, 1)$ over the tail probability t, weighted by the Beta$(\alpha, 1)$ density $\alpha t^{\alpha-1}$:

$$g_\alpha(s) = \int_0^1 \min(s/t, 1)\, \alpha t^{\alpha-1}\, dt = \frac{\alpha s - s^\alpha}{\alpha - 1},$$

with $g_1(s) = s(1 - \log s)$ the limit at $\alpha = 1$. It has no jump at zero, and it puts no weight on the mean or the maximum alone. Its Gini index, $2\int_0^1 g - 1$ (see Using distortions), is $1/(\alpha+1)$, so the member with index G is $\alpha = 1/G - 1$. The family runs from the maximum as $\alpha \to 0$ to the mean as $\alpha \to \infty$, and $\alpha = 2$ is the dual distortion $1 - (1-s)^2$.

The name comes from the price. Since $g_\alpha(S) = S + (S - S^\alpha)/(\alpha - 1)$,

$$g_\alpha[X] = \mathsf{P}X + \int_0^\infty \frac{S(x) - S(x)^\alpha}{\alpha - 1}\, dx,$$

the expected loss plus the cumulative residual Tsallis entropy of X of order α. At $\alpha = 1$ the loading is the cumulative residual entropy $-\int_0^\infty S \log S\, dx$. The family is used as a sixth pricing family beside the five standard ones. Like them it has one parameter and no jump at zero, and its slope at zero is $\alpha/(\alpha - 1)$ for $\alpha > 1$, finite where PH and Wang have an infinite slope.

```
distortion NAME [as "label"] tsallis alpha
distortion T tsallis 3                          # index 1/4
distortion Half as "Tsallis half" tsallis 0.5   # g(s) = 2 sqrt(s) - s, index 2/3
```

```
import numpy as np
import pandas as pd
from aggregate import build, Distortion, __version__
from aggregate.spectral import consistent_distortions
from style import wprint, GT, refuse, figures, figure
figures()
print(__version__)
```

```
2.1.0a5
```

[1] From aggregate 2.1.0a3.

Tsallis distortions from DecL and from Python

The DecL statement `distortion` with the kind `tsallis` builds a Tsallis distortion. In Python the factory `Distortion('tsallis', alpha)`, the static method `Distortion.tsallis` and the constructor `Distortion.from_gini('tsallis', 0.25)` build the same object. All four below have the label `Tsallis(3)`, the index 0.25 and the same `id`.

```
t = build('distortion T tsallis 3')
for d in [t, Distortion('tsallis', alpha=3.0), Distortion.tsallis(3.0),
          Distortion.from_gini('tsallis', 0.25)]:
    print(f'{type(d).__name__:18s} {d.label:10s} {d.gini:.4f} {d.id()}')
```

```
TsallisDistortion  Tsallis(3) 0.2500 912DAE9A
TsallisDistortion  Tsallis(3) 0.2500 912DAE9A
TsallisDistortion  Tsallis(3) 0.2500 912DAE9A
TsallisDistortion  Tsallis(3) 0.2500 912DAE9A
```

The attribute `info` summarizes the distortion. Its `weights mean` and `weights max` rows are both 0, and its TVaR mixture has no atoms inside $(0, 1)$, since the Beta mixture is continuous.

```
wprint(t.info)
```

```
distortion object name   tsallis
kind                     tsallis
kind name                tsallis (beta(alpha, 1) mixture of tvars)
shape                    3
shape name               alpha
other params             none
weights mean             0
weights max              0
interior atoms           False
gini                     0.25
area                     0.625
id                       912DAE9A
```

The Tsallis distortion beside the standard families

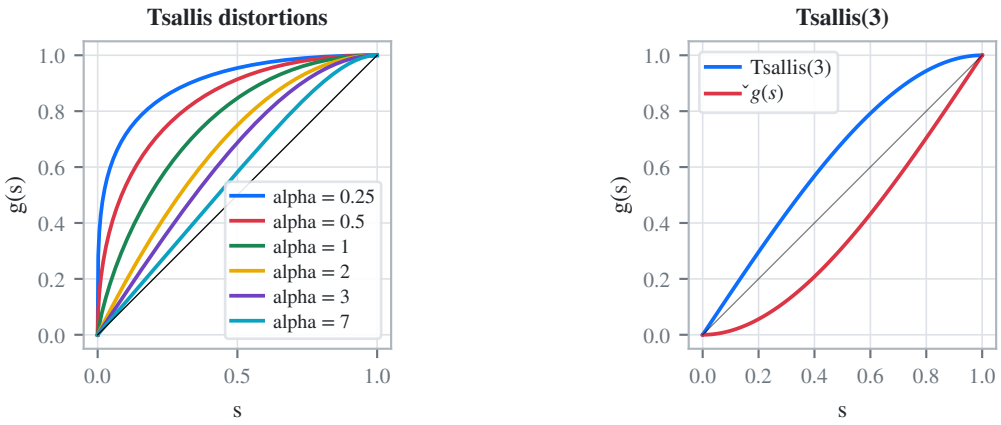
The function `consistent_distortions(p, extended=True)` in `aggregate.spectral` returns the five standard families and `tsallis`, all with index p. The table shows $g(s)$ at five values of s for index 0.25. At $s = 0.001$, an event in the extreme tail, only TVaR gives less weight than Tsallis, 0.0013 against 0.0015, and at $s = 0.5$ Tsallis gives the most, 0.6875. Of the six it is closest to dual, which gives 0.0017 at $s = 0.001$ and 0.6850 at $s = 0.5$.

```
fams = consistent_distortions(0.25, extended=True)
ss = [0.001, 0.01, 0.1, 0.5, 0.9]
GT(pd.DataFrame([[d.gini, d.label, *d.g(np.array(ss))] for d in fams.values()],
                columns=['gini', 'label', *[f'g({x})' for x in ss]],
                index=pd.Index(list(fams), name='kind')), float_format='.4f')
```

kind	gini	label	g(0.001)	g(0.01)	g(0.1)	g(0.5)	g(0.9)
ccoc	0.2500	ccoc(0.333)	0.2507	0.2575	0.3250	0.6250	0.9250
ph	0.2500	PH(0.6)	0.0158	0.0631	0.2512	0.6598	0.9387
wang	0.2500	Wang(0.451)	0.0042	0.0303	0.2030	0.6739	0.9584
tsallis	0.2500	Tsallis(3)	0.0015	0.0150	0.1495	0.6875	0.9855
dual	0.2500	dual(1.67)	0.0017	0.0166	0.1610	0.6850	0.9785
tvar	0.2500	TVaR(0.25)	0.0013	0.0133	0.1333	0.6667	1.0000

The left panel shows g_α for six orders, from $\alpha = 0.25$, index 0.8, to $\alpha = 7$, index 0.125. The right panel is `Distortion.plot` for $\alpha = 3$, which draws g and its dual $\check{g}$.

```
s = np.linspace(0, 1, 1001)
fig, (ax0, ax1) = figure('full', cols=2)
for alpha in [0.25, 0.5, 1, 2, 3, 7]:
    ax0.plot(s, Distortion.tsallis(alpha).g(s), label=f'alpha = {alpha}')
ax0.plot([0, 1], [0, 1], c='k', lw=0.5)
ax0.set(xlabel='s', ylabel='g(s)', aspect='equal', title='Tsallis distortions')
ax0.legend(fontsize='small')
t.plot(ax=ax1);
```



Calibrating the extended set

The method `calibrate_distortions` of an aggregate or a portfolio calibrates the five standard families by default, and with `extended=True` it adds `tsallis` between `wang` and `dual`. The table shows the six families fitted to a 10% cost of capital at the 99% VaR of a book with 50 expected claims and a lognormal severity with mean 40 and CV 1.8 (see Using `price` and `evaluate` for the reverse question). The Tsallis order is solved for in its index, and the fitted $\alpha = 4.475$ has index 0.183, between dual at 0.175 and TVaR at 0.185. The five standard rows are the same as a default calibration gives.

```
book = build('agg Book 50 claims sev lognorm 40 cv 1.8 poisson')
res = book.calibrate_distortions(0.1, p=0.99, extended=True)
GT(res.distortion_df[['gini', 'param_name', 'param', 'error']],
   formatters={'gini': '{x:.4f}', 'param': '{x:.4g}'})
```

distortion	gini	param_name	param	error
ccoc	0.0909	r	0.1	0.00e+00
ph	0.1392	a	0.7555	8.88e-10
wang	0.1581	lam	0.2821	2.60e-07
tsallis	0.1826	alpha	4.475	0.00e+00
dual	0.1752	b	1.425	-8.71e-09
tvar	0.1853	p	0.1853	7.44e-06

The price at order one and the cumulative residual entropy

The price of the book under g_1 is its expected loss plus the cumulative residual entropy $-\int S \log S \, dx$. Both sides, computed from the survival function of the book on its grid, agree to rounding:

```
S = book.density_df.S.to_numpy()
el = S.sum() * book.bs
cre = -(S[S > 0] * np.log(S[S > 0])).sum() * book.bs
print(f'g_1[X] = {Distortion.tsallis(1).g(S).sum() * book.bs:,.4f}, '
      f'P X + CRE = {el:,.4f} + {cre:,.4f} = {el + cre:,.4f}')
```

```
g_1[X] = 2,566.8986, P X + CRE = 1,999.9998 + 566.8988 = 2,566.8986
```

Refused calls

Four calls are refused: an order that is not positive, a Tsallis distortion with two parameters in DecL, `calibrate_distortions` with both `names` and `extended=True`, which choose the same thing, and `Distortion.from_gini` with an index outside $(0, 1)$, which no Tsallis distortion reaches.

```
refuse(lambda: Distortion('tsallis', alpha=0),
       'distortion T tsallis 3 4',
       lambda: book.calibrate_distortions(0.1, p=0.99, names=['ph'], extended=True),
       lambda: Distortion.from_gini('tsallis', 1.0))
```

```
TsallisDistortion: alpha must be finite and positive, got 0
DecL distortion 'tsallis' expects 1 parameter(s) ('alpha',), got 2: [3.0, 4.0]
calibrate_distortions takes names= or extended=True, not both: extended=True is
the families ('ccoc', 'ph', 'wang', 'tsallis', 'dual', 'tvar').
from_gini needs a Gini index in the open interval (0, 1), got 1.0.
```