Using distortions

Distortion functions for risk-adjusted pricing with the DecL distortion statement and the Distortion class

Stephen J. Mildenhall

2026-11-06

A **distortion** is an increasing function $g : [0,1] \to [0,1]$ with $g(0) = 0$ and $g(1) = 1$. It prices a loss $X \geq 0$ with survival function $S(x) = \Pr(X > x)$ as

$$g[X] = \int_0^\infty g(S(x))\,dx,$$

which is the expected loss $\mathsf{P}X = \int_0^\infty S(x)\,dx$ when $g(s) = s$. Square brackets mark g acting on a random variable, as a spectral risk measure, and parentheses mark g acting on a probability. A pricing distortion is concave, so $g(s) \geq s$: it moves probability toward the large losses and $g[X] \geq \mathsf{P}X$. Its **dual** $\check{g}(s) = 1 - g(1-s)$ is convex, and $\check{g}[X] \leq \mathsf{P}X$ is the bid price. Every concave distortion is a mixture of TVaR distortions, so $g[X]$ is a weighted average of the TVaRs of X at different levels.

The five families of the standard pricing set each have one shape parameter.

kind	$g(s)$	**parameter**	gini
`ccoc`	$d + (1-d)s$ for $s > 0$	return r, with $d = r/(1+r)$	d
`ph`	s^a	$0 < a \leq 1$	$(1-a)/(1+a)$
`wang`	$\Phi(\Phi^{-1}(s) + \lambda)$	$\lambda \geq 0$	$2\Phi(\lambda/\sqrt{2}) - 1$
`dual`	$1 - (1-s)^b$	$b \geq 1$	$(b-1)/(b+1)$
`tvar`	$\min(s/(1-p), 1)$	$0 \leq p \leq 1$	p

The constant cost of capital `ccoc` has a jump of d at zero, and with assets a it prices $X \wedge a$ at $(1-d)\,\mathsf{P}(X \wedge a) + d\,a$, the premium that earns the return r on the capital. Other important distortions include: `tsallis`, the Beta$(\alpha, 1)$ mixture of TVaRs, $g(s) = (\alpha s - s^\alpha)/(\alpha - 1)$, introduced in Using `tsallis`; `bitvar`, a mixture of TVaRs at $p_0 < p_1$ with weight w_1 on p_1; `wtdtvar`, a weighted TVaR at several levels, built in Python; `beta`; and `cll`, `clin`, `lep` and `ly`, which have a second parameter, the intercept `r0`. The combinators `minimum` and `mixture` take other distortions as their arguments.

The index $\text{gini} = 2\int_0^1 g(s)\,ds - 1$ is twice the area between g and the diagonal, also equal to twice the margin on a standard uniform. A TVaR at level p has gini $= p$, so the index is the level of the TVaR with the same area. It puts families with incomparable parameters on one scale that increases from 0 for risk neutral to 1 for max pricing. All the tables of distortions on these pages lead with it. The attribute `gini` holds the closed form where a family has one and otherwise integrates g numerically, so every distortion has an index.

```
distortion NAME [as "label"] kind n1 [n2 ...]
distortion Gentle ph 0.6                          # g(s) = s^0.6
distortion Named as "my distortion" ph 0.6        # the same, with a declared label
distortion Cost ccoc 0.1                          # a 10% return on capital
distortion Two bitvar 0.5 0.99 0.25               # p0, p1, then the weight on p1
distortion Low minimum dist.Gentle dist.Two       # the pointwise minimum
distortion Mix mixture dist.Gentle dist.Two wts [0.6 0.4]
```

The numbers after the kind are its parameters in a fixed order, and for `ccoc` the number is the return r. A reference `dist.NAME` copies a stored distortion into a combinator. An `as` clause after the name declares the label, and the attribute `name` stays the family. Pricing with a distortion is the subject of Using `price`, and finding the distortion that a premium implies is the subject of Using `price` and `evaluate`.

```
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
```

Distortions from DecL and from Python

The DecL statement `distortion` returns a `Distortion`, and the factory `Distortion(kind, shape)` builds the same object in Python. The shape can also be passed by its natural name: `a` for `ph`, `lam` for `wang`, `b` for `dual` and `p` for `tvar`. The static methods `Distortion.ph`, `Distortion.wang` and so on are shortcuts. All four forms below give the same distortion, with the same label `PH(0.6)`, the same index 0.25 and the same `id`.

```
gentle = build('distortion Gentle ph 0.6')
for d in [gentle, Distortion('ph', 0.6), Distortion('ph', a=0.6), Distortion.ph(0.6)]:
    print(f'{type(d).__name__:14s} {d.label:8s} {d.gini:.4f} {d.id()}')
print(gentle.program)
```

```
PHDistortion   PH(0.6)  0.2500 E83B5ED9
PHDistortion   PH(0.6)  0.2500 E83B5ED9
PHDistortion   PH(0.6)  0.2500 E83B5ED9
PHDistortion   PH(0.6)  0.2500 E83B5ED9
distortion Gentle ph 0.6
```

An `as` clause changes the label and leaves the family in `name`:

```
named = build('distortion Named as "my distortion" ph 0.6')
print(named.name, '|', named.label, '|', named.program)
```

```
ph | my distortion | distortion Named as "my distortion" ph 0.6
```

The shortcut for `ccoc` takes the return `r=`, as DecL does, or the discount `d=`, with $d = r/(1+r)$, always by keyword. All four forms below give $d = 1/11$, a 10% return.

```
print(build('distortion Cost ccoc 0.1').d, Distortion('ccoc', r=0.1).d,
      Distortion.ccoc(r=0.1).d, Distortion.ccoc(d=1 / 11).d)
```

```
0.09090909090909091 0.09090909090909091 0.09090909090909091 0.09090909090909091
```

The attribute `info` summarizes a distortion. Its `weights mean` and `weights max` rows are the weights its TVaR mixture puts on the mean ($p = 0$) and on the maximum ($p = 1$). A PH distortion puts weight a on the mean, here 0.6, and none on the maximum; `area` is $\int_0^1 g = (1 + \text{gini})/2$.

```
wprint(gentle.info)
```

```
distortion object name   ph
kind                     ph
kind name                proportional hazard
shape                    0.6
shape name               a
other params             none
weights mean             0.6
weights max              0
interior atoms           False
gini                     0.25
area                     0.625
id                       E83B5ED9
```

Distortions in the recipe library

The recipe library included with aggregate (see Using recipes) holds one distortion of each standard family, each with gini $= 0.2$, and a `minimum` and a `mixture` of two of them. A bare name builds the stored entry. The column `2∫g − 1` is the index computed from g on a fine grid, and it agrees with `gini` for all seven. The index is linear in g, so the mixture of two distortions with index 0.2 has index 0.2, and the minimum lies below both and has a smaller index, 0.162.

```
s = np.linspace(0, 1, 100_001)
lib = {n: build(n) for n in build.discover(kind='distortion', tags='topic:spectral').index}
GT(pd.DataFrame({'gini': [d.gini for d in lib.values()],
                 '2∫g − 1': [2 * np.trapezoid(d.g(s), s) - 1 for d in lib.values()],
                 'label': [d.label for d in lib.values()]},
                index=pd.Index(list(lib), name='recipe')))
```

recipe	gini	2∫g − 1	label
CCoCDistortion	0.2000	0.2000	ccoc(0.25)
DualDistortion	0.2000	0.2000	dual(1.5)
MinimumDistortion	0.1623	0.1623	minimum(2)
MixtureDistortion	0.2000	0.2000	mixture(2)
PHDistortion	0.2000	0.2000	PH(0.667)
TVaRDistortion	0.2000	0.2000	TVaR(0.2)
WangDistortion	0.2000	0.2000	Wang(0.358)

Five families with one Gini index

The function `consistent_distortions(p)` in `aggregate.spectral` returns one distortion of each standard family, all with gini $= p$. They enclose the same area and differ in where they put it. The table shows $g(s)$ at five values of s. At $s = 0.001$, an event in the extreme tail, `ccoc` gives weight 0.251 and PH 0.016, while dual and TVaR give less than 0.002. At $s = 0.5$ `ccoc` gives the least weight, 0.625, and dual the most, 0.685. For the same index, the tail-heavy families `ccoc` and PH load a rare large loss more heavily, and dual and TVaR load the body of the distribution.

```
fams = consistent_distortions(0.25)
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')))
```

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
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 constructor `Distortion.from_gini(kind, gini)` builds the member of a one-parameter family with a given index, for `ccoc`, `ph`, `wang`, `tsallis`, `dual` and `tvar`, and `consistent_distortions` uses the same inverses. Each distortion below has index 0.25:

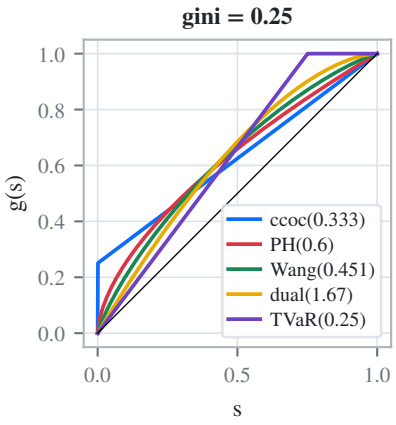
```
for kind in ['ph', 'wang', 'tsallis', 'dual']:
    d = Distortion.from_gini(kind, 0.25)
    print(f'{d.label:12s} {d.gini:.6f}')
```

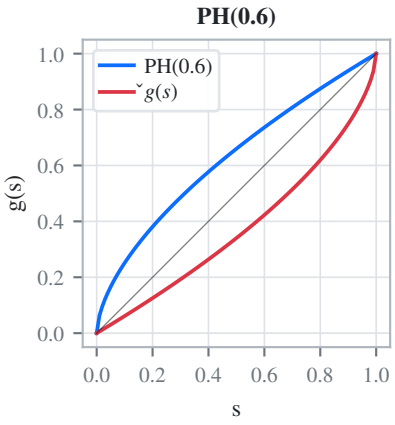
```
PH(0.6)      0.250000
Wang(0.451)  0.250000
Tsallis(3)   0.250000
dual(1.67)   0.250000
```

Plotting a distortion

The left panel shows the five distortions with gini $= 0.25$. The `ccoc` line starts at $d = 0.25$, the jump at zero, and TVaR reaches 1 at $s = 1 - p = 0.75$. The right panel is `Distortion.plot`, which draws g and, by default, its dual $\check{g}$, on axes of equal length, since the concavity of g is what the plot is for.

```
fig, (ax0, ax1) = figure('full', cols=2)
for d in fams.values():
    ax0.plot(s, d.g(s), label=d.label)
ax0.plot([0, 1], [0, 1], c='k', lw=0.5)
ax0.set(xlabel='s', ylabel='g(s)', aspect='equal', title='gini = 0.25')
ax0.legend(fontsize='small')
gentle.plot(ax=ax1);
```





Refused programs

Four distortion programs are refused: `wtdtvar`, which has no DecL form because its parameters are vectors; a `ph` with two parameters; a `bitvar` with $p_0 > p_1$; and a kind that does not exist. The last message lists the available kinds. In Python, `Distortion.ccoc` refuses a bare number, which could be either the return or the discount, and `Distortion.from_gini` refuses a family with more than one parameter, listing its parameters.

```
refuse('distortion W wtdtvar 0.5 0.9',
       'distortion P ph 0.6 0.7',
       'distortion B bitvar 0.9 0.5 0.5',
       'distortion X expo 0.5',
       lambda: Distortion.ccoc(0.1),
       lambda: Distortion.from_gini('bitvar', 0.25))
```

```
Distortion 'wtdtvar' has no DecL number-list form; construct it in Python
(weighted-TVaR takes parameter vectors; the minimum/mixture combinators take
distortion references).
DecL distortion 'ph' expects 1 parameter(s) ('a',), got 2: [0.6, 0.7]
bitvar requires p0 < p1, got p0=0.9, p1=0.5
Unknown distortion kind 'expo'; available: ['beta', 'bitvar', 'ccoc', 'clin',
'cll', 'dual', 'lep', 'ly', 'minimum', 'mixture', 'ph', 'power', 'tsallis',
'tvar', 'wang', 'wtdtvar']
Distortion.ccoc() takes 0 positional arguments but 1 was given
from_gini builds a one-parameter family; 'bitvar' takes parameters ('p0', 'p1',
'w1'). The one-parameter families are ccoc, ph, wang, tsallis, dual, tvar.
```