Using oep

Occurrence exceeding probability and occurrence PML of a Poisson aggregate, compared with aggregate PML (oep)

Stephen J. Mildenhall

2026-11-07

The **occurrence exceeding probability** (OEP) of a loss x is the probability that the largest claim in a year exceeds x, and the loss at a given OEP is an **occurrence PML** (probable maximum loss). The **aggregate exceeding probability** (AEP) is the probability that the year's total loss exceeds x, and the loss at a given AEP is a quantile of the aggregate distribution, the aggregate PML. The total is at least the largest claim, so at every probability the aggregate PML is at least the occurrence PML.

With a Poisson claim count of mean λ and severity survival function $S(x) = \Pr(X > x)$, the claims exceeding x are Poisson with mean $\lambda S(x)$, by thinning. A year has none of them with probability $e^{-\lambda S(x)}$, so

$$\mathrm{OEP}(x) = \Pr(\max_i X_i > x) = 1 - \exp(-\lambda S(x)),$$

and the occurrence PML at annual probability p is $x = S^{-1}(-\log(1-p)/\lambda)$. For a general count with probability generating function $\mathcal{P}_N$, $\Pr(\max_i X_i \le x) = \mathcal{P}_N(1 - S(x))$, and the Poisson formula is the case $\mathcal{P}_N(z) = e^{-\lambda(1-z)}$.

The function `oep(agg, p)`, in `aggregate.utilities` and importable from `aggregate`, computes the occurrence PML of a Poisson aggregate at one or more probabilities p. It returns a table indexed by p with the loss, its severity exceedance `S_sev` and percentile `F_sev`, the achieved OEP, and two return periods: the occurrence return period $1/(\lambda S(x))$, the mean time between claims exceeding x, and the annual return period $1/\mathrm{OEP}(x)$, the mean time between years containing such a claim. The loss is computed from `Aggregate.sev`, the exact continuous severity, so it is not rounded to the grid, and the aggregate need not be updated. The argument `freq` replaces λ. Any count other than a plain Poisson is refused, since the formula depends on thinning.

```
import numpy as np
import pandas as pd
from aggregate import build, oep, __version__
from style import GT, refuse, figures
figures()
print(__version__)
```

```
2.1.0a7
```

Occurrence and aggregate PMLs of a catastrophe book

The example book has two events a year on average, each a single claim, with a lognormal severity of median 1,000 and shape $\sigma = 1$, so mean 1,648.7. The table compares the occurrence PML from `oep` with the aggregate PML, the quantile `q(1 - p)` of the aggregate, at the same annual probabilities.

```
a = build('agg Cat 2 claims sev 1000 * lognorm 1 poisson')
ps = [0.5, 0.2, 0.1, 0.04, 0.02, 0.01, 0.004, 0.001]
o = oep(a, ps)
o['AEP loss'] = [a.q(1 - p) for p in ps]
o['ratio'] = o['AEP loss'] / o['loss']
o.index = pd.Index([f'{p:g}' for p in ps], name='p')
GT(o[['loss', 'occurrence_return_period', 'annual_return_period',
      'AEP loss', 'ratio']], float_format=',.2f')
```

p	loss	occurrence_return_period	annual_return_period	AEP loss	ratio
0.5	1,483.77	1.44	2.00	2,180	1.47
0.2	3,381.14	4.48	5.00	5,330	1.58
0.1	5,050.08	9.49	10.00	7,800	1.54
0.04	7,731.74	24.50	25.00	11,320	1.46
0.02	10,201.78	49.50	50.00	14,270	1.40
0.01	13,119.41	99.50	100.00	17,520	1.34
0.004	17,770.32	249.50	250.00	22,420	1.26
0.001	26,853.23	999.50	1,000.00	31,560	1.18

At 1 in 100 the occurrence PML is 13,119 and the aggregate PML 17,520. Among the probabilities shown, the ratio of the two is largest, 1.58, at 1 in 5, and it falls to 1.18 at 1 in 1,000: the lognormal is subexponential, so a large annual total is usually the result of one large event. The aggregate PMLs are multiples of the bucket size, 10, and the occurrence PMLs are exact. For a continuous severity the achieved `oep` column, omitted here, equals p.

The two return periods differ most at common losses. At $p = 1/2$ a claim above 1,484 occurs every 1.44 years on average, but a year containing one occurs every 2 years, because some years have two or more. For rare losses, with $u = \lambda S(x)$ small, $1/(1-e^{-u}) = 1/u + 1/2 + O(u)$, so the annual return period exceeds the occurrence return period by half a year, 1,000 against 999.5 at 1 in 1,000.

The occurrence exceeding probability as a layer probability

OEP(x) is the probability that an unlimited occurrence layer `inf xs x` pays anything in the year (see Using occurrence reinsurance). The probability is one minus the mass at zero of the ceded aggregate, and `aggregate` computes it for any claim count. The table compares it with the formula for the Poisson count and for a negative binomial count, `mixed gamma 0.5`, a Poisson with a gamma mixing distribution of CV $c = 0.5$, whose generating function is $\mathcal{P}_N(z) = (1 + c^2\lambda(1-z))^{-1/c^2}$.

```
layer = 'agg L 2 claims sev 1000 * lognorm 1 occurrence ceded to inf xs {} {}'
rows, c2 = {}, 0.5 ** 2
for x in [5000, 10000, 20000]:
    s = a.sev.sf(x)
    pr = {m: 1 - build(layer.format(x, m), bs=1, log2=18).density_df.p_total.iloc[0]
          for m in ['poisson', 'mixed gamma 0.5']}
    rows[x] = [pr['poisson'], -np.expm1(-a.n * s),
               pr['mixed gamma 0.5'], 1 - (1 + c2 * a.n * s) ** (-1 / c2)]
df = pd.DataFrame.from_dict(rows, orient='index', columns=[
    'Poisson layer', 'Poisson formula', 'mixed layer', 'mixed formula'])
df.index.name = 'x'
GT(df, float_format='.6f')
```

x	Poisson layer	Poisson formula	mixed layer	mixed formula
5,000	0.101922	0.101942	0.100647	0.100666
10,000	0.021074	0.021077	0.021019	0.021022
20,000	0.002734	0.002734	0.002733	0.002733

The layer and the formula agree to within 0.02% (relative), the error of a grid with buckets of width 1. Mixing lowers the OEP at every x, by Jensen's inequality, since $1 - \mathsf{P}e^{-\Lambda S(x)} \le 1 - e^{-\lambda S(x)}$ for a random Poisson mean Λ with $\mathsf{P}\Lambda = \lambda$. The function `oep` raises an error on the mixed book, and the layer probability is the way to compute its OEP.

Limited severities and the freq argument

On a book with a policy limit the severity has an atom at the limit, and every probability below the annual probability of a claim at the limit gives the limit as the occurrence PML. With a limit of 20,000 that probability is $1 - \exp(-2S(20000)) = 0.002734$. The 1 in 1,000 PML is therefore 20,000, and the achieved `oep`, the probability of a claim strictly above it, is zero, so its return period is infinite (shown as a dash). The 1 in 100 PML is 13,119, the same as without the limit. Passing `freq=4` doubles the event rate, which raises the 1 in 100 PML to 16,534.

```
b = build('agg Lim 2 claims 20000 xs 0 sev 1000 * lognorm 1 poisson')
df = pd.concat({'limit 20,000': oep(b, [0.01, 0.001]),
                'Cat, freq=4': oep(a, 0.01, freq=4)}, names=['book'])
GT(df[['loss', 'oep', 'annual_return_period']], float_format=',.4f')
```

book	*p*	loss	oep	annual_return_period
limit 20,000	0.0100	13,119.4077	0.0100	100.0000
	0.0010	20,000.0000	0.0000	—
Cat, freq=4	0.0100	16,533.9524	0.0100	100.0000

An occurrence program

On an aggregate with an `occurrence net of` or `occurrence ceded to` clause, `oep` returns net or ceded points.[1] The net and ceded losses of a claim are nondecreasing functions of its gross loss, so the largest net or ceded claim in a year is that function of the largest gross claim, and the point is the function applied to the gross point. The example places 10,000 xs 10,000 on the catastrophe book. At 1 in 100 the gross point is 13,119, the net point is the retention of 10,000, and the ceded point is 3,119. Where a function is constant, its loss has an atom, and the achieved OEP falls below p. The net loss is 10,000 for every claim between 10,000 and 20,000, so the 1-in-100 net point is exceeded only by a claim above 20,000, once in 366 years. The ceded loss is 0 below 10,000, so a 1-in-10 ceded point of 0 is exceeded once in 47 years, and the ceded loss is 10,000 above 20,000, so the 1-in-1,000 ceded point is never exceeded.

[1] From aggregate 2.1.0a7

```
df = pd.concat({kind: oep(build(f'agg CatRe 2 claims sev 1000 * lognorm 1 '
                                f'occurrence {kind} 10000 xs 10000 poisson',
                                update=False), [0.1, 0.01, 0.001])
                for kind in ['net of', 'ceded to']}, names=['program'])
GT(df[['loss', 'oep', 'annual_return_period']], float_format=',.4f')
```

program	*p*	loss	oep	annual_return_period
net of	0.1000	5,050.0761	0.1000	10.0000
	0.0100	10,000.0000	0.0027	365.7481
	0.0010	16,853.2270	0.0010	1,000.0000
ceded to	0.1000	0.0000	0.0211	47.4453
	0.0100	3,119.4077	0.0100	100.0000
	0.0010	10,000.0000	0.0000	—

An event loss table

An aggregate built from an event loss table with secondary uncertainty, an `elt` clause with six-column rows and no `no2u` flag (see Using elt), has a severity that is the rate-weighted mixture of the events' beta and two-point laws. The eltq package computes its severity functions exactly from the rows,[2] so `oep` applies to it as to any Poisson aggregate. The table compares the occurrence PMLs of a six-event table with and without secondary uncertainty. With it, the achieved OEP equals p. Without it (`no2u`), each event loss is an atom at its mean, so every occurrence PML is one of the six means and the achieved OEP falls below p. At 1 in 250 and rarer the PML is the largest mean, 250, which no claim exceeds, so the achieved OEP is zero.

```
table = '''[[1 0.1 5 50 4 2] [2 0.08 12 120 8 5] [3 0.05 25 250 15 10]
           [4 0.03 60 400 30 20] [5 0.02 120 800 50 40]
           [6 0.01 250 1500 100 80]]'''
ps = [0.1, 0.04, 0.01, 0.004, 0.001]
df = pd.concat({flag or 'secondary uncertainty':
                oep(build(f'agg Cat6 elt {flag} {table}', update=False), ps)
                for flag in ['', 'no2u']}, names=['model'])
GT(df[['loss', 'oep', 'annual_return_period']], float_format=',.4f')
```

model	*p*	loss	oep	annual_return_period
secondary uncertainty	0.1000	16.5314	0.1000	10.0000
	0.0400	59.2989	0.0400	25.0000
	0.0100	190.8643	0.0100	100.0000
	0.0040	307.7952	0.0040	250.0000
	0.0010	506.3936	0.0010	1,000.0000
no2u	0.1000	25.0000	0.0582	17.1717
	0.0400	60.0000	0.0296	33.8358
	0.0100	120.0000	0.0100	100.5008
	0.0040	250.0000	0.0000	—
	0.0010	250.0000	0.0000	—

Each evaluation of the severity is a pass over every row of the table, so on a table of a million rows `oep` at 30 probabilities takes a few seconds. The aggregate need not be updated. When it is, the

[2] From aggregate 2.1.0a6

severity quantile on the grid, `q_sev(1 - s)` with $s = -\log(1-p)/\lambda$, is within half a bucket of the exact loss: with buckets of 1/8 it gives 190.875 at 1 in 100, against 190.864 from `oep`.

```
c = build(f'agg Cat6 elt {table}')
s = -np.log1p(-np.array(ps)) / c.n
print(c.bs, [float(c.q_sev(1 - v)) for v in s])
```

```
0.125 [16.5, 59.25, 190.875, 307.75, 506.375]
```

Refused calls

The probability p must lie strictly between 0 and the probability of at least one claim, $1 - e^{-\lambda}$, which is 0.8647 for $\lambda = 2$, since a year with no claims has no largest claim. A negative binomial or a zero-modified Poisson count is refused.

```
nb = build('agg NB 2 claims sev 1000 * lognorm 1 mixed gamma 0.5')
zm = build('agg ZM 2 claims sev 1000 * lognorm 1 poisson zm 0.2')
refuse(lambda: oep(a, 0.9), lambda: oep(a, 0),
       lambda: oep(nb, 0.01), lambda: oep(zm, 0.01))
```

```
p must be below the ceiling 1 - exp(-lam) = 0.8646647 for lam = 2; got [0.9]. A
year with no occurrence has no largest loss, so nothing is exceeded.
p must be in (0, 1) exclusive; got [0.0].
oep requires Poisson frequency; NB has freq_name='gamma'.
oep requires plain Poisson frequency; ZM is zero modified, which breaks the 1 -
exp(-lam S) derivation.
```