Using `zm` and `zt`

Zero-truncated and zero-modified claim counts, and the two geometric distributions

Stephen J. Mildenhall

2026-11-01

A zero-modified count resets the probability of no claims and rescales the rest. If N has $p_k = \Pr(N = k)$, the modified count N^M has $\Pr(N^M = 0) = p_0^M$ and $\Pr(N^M = k) = c\, p_k$ for $k \geq 1$, with $c = (1 - p_0^M)/(1 - p_0)$, so $\mathsf{P}N^M = c\,\mathsf{P}N$. Zero truncation is the case $p_0^M = 0$. The modified counts form the (a, b, 1) class. They are used to fit a count whose probability of no claims differs from that of its family. In DecL the suffix `zt` after the frequency truncates the count, and `zm 0.5` modifies it to have $p_0^M = 1/2$. Both apply to the Poisson, binomial, negative binomial, geometric, and logarithmic counts.

The modification moves the mean, so the claim count in the exposure clause could set either the realized mean $\mathsf{P}N^M$ or the base mean $\mathsf{P}N$. The claim count sets the realized mean: `4 claims ... poisson zt` has a mean of four claims, and the base mean is solved for. A trailing `!` gives the textbook reading instead, where the clause sets the base mean and the realized mean is an output.

The geometric distribution has two conventions. The default `geometric` counts failures before the first success, on 0, 1, 2, ..., with $\Pr(G = k) = p(1 - p)^k$ and mean $(1 - p)/p$, so a mean of n gives $p = 1/(n + 1)$. The number of trials T, on 1, 2, ..., has $\Pr(T = k) = p(1 - p)^{k-1}$ and mean $1/p$. By memorylessness $G \mid G \geq 1$ has the law of $1 + G$, which is the law of T. The trials geometric is therefore the zero-truncated failures geometric, and it is written `geometric zt`.

```
agg PoZT 4 claims dsev [1] poisson zt          # mean 4, P(N = 0) = 0
agg PoZM 4 claims dsev [1] poisson zm 0.5      # mean 4, P(N = 0) = 1/2
agg PoBase 4 claims dsev [1] poisson zt !      # base mean 4, mean 4.0746
agg Trials 4 claims dsev [1] geometric zt      # trials, p = 1/4, mean 4
```

```
import warnings
import numpy as np
import pandas as pd
from scipy import stats
from aggregate import build, qd, __version__
from style import wprint, GT, refuse, figures
figures()
print(__version__)
```

```
1.60.0
```

With severity `dsev [1]` the aggregate is the count itself, so `density_df.p_total` is the probability mass function of N. A helper tabulates $\Pr(N = 0)$, $\Pr(N = 1)$, the realized mean `n`, the base mean, and the mean of the computed distribution. The grid is given explicitly, since the automatic one is too short for a geometric tail.

```
def counts(*clauses, n=4):
    rows = {}
    for clause in clauses:
        a = build(f'agg Count {n} claims dsev [1] {clause}', bs=1, log2=8)
        p = a.density_df.p_total
        rows[clause] = [p[0], p[1], a.n, a.base_mean, a.est_m]
    return pd.DataFrame.from_dict(rows, orient='index', columns=[
        'P(N=0)', 'P(N=1)', 'mean', 'base mean', 'computed mean']
        ).rename_axis('clause')
```

Zero-truncated and zero-modified Poisson counts

For a Poisson count with four claims in the exposure clause, a bare `zt` or `zm` keeps the mean at four and solves for the base mean. With `!` the base mean is four and the realized mean moves instead, up under truncation and down when the probability of no claims rises to 1/2. Keeping a mean of four claims with half the outcomes at zero needs a base mean near 8.

```
po = counts('poisson', 'poisson zt', 'poisson zt !', 'poisson zm 0.5',
            'poisson zm 0.5 !')
GT(po)
```

clause	P(N=0)	P(N=1)	mean	base mean	computed mean
poisson	0.0183	0.0733	4.0000	4.0000	4.0000
poisson zt	0.0000	0.0793	4.0000	3.9207	4.0000
poisson zt !	0.0000	0.0746	4.0746	4.0000	4.0746
poisson zm 0.5	0.5000	0.0013	4.0000	7.9973	4.0000
poisson zm 0.5 !	0.5000	0.0373	2.0373	4.0000	2.0373

The zero-truncated Poisson with base mean λ has mean $\lambda/(1-e^{-\lambda})$. The base mean 3.9207 solved for by `poisson zt` gives a mean of 4, and the base mean 4 of `poisson zt !` gives 4.0746, as in the table.

```
for lam in [po.loc['poisson zt', 'base mean'], 4]:
    print(f'{lam:.6f} {lam / (1 - np.exp(-lam)):.6f}')
```

```
3.920690 4.000000
4.000000 4.074629
```

Zero modification with a loss or premium exposure

The exposure clause of a zero-modified count can also be an expected loss or a premium, such as `400 loss`. A bare `zm` matches the stated loss: four claims of 100 give 400. With `!` the mean count moves away from the target, to 2.0373 claims and a loss of 203.73, and a `ZeroModifiedExposureWarning` gives the loss delivered.

```
with warnings.catch_warnings(record=True) as caught:
    warnings.simplefilter('always')
    for program in ['agg Loss 400 loss dsev [100] poisson zm 0.5',
                    'agg Loss 400 loss dsev [100] poisson zm 0.5 !']:
        a = build(program, bs=100, log2=8)
        print(f'{a.n:.4f} {a.est_m:.2f}')
for w in caught:
    wprint(f'{w.category.__name__}: {w.message}')
```

```
4.0000 400.00
2.0373 203.73
ZeroModifiedExposureWarning: Loss: the exposure clause states a monetary target
of 400, but zm/zt shifts the mean off it: E[N] 4 -> 2.03731 delivers 203.731.
Drop the ! from the zm/zt clause to pin the target instead.
```

Failures and trials geometric counts

The default `geometric`, counting failures, with mean 4 has $p = 1/5$, so $\Pr(N = 0) = 1/5$. The count `geometric zt` with mean 4 is the trials geometric with $p = 1/4$: $\Pr(N = 1) = 1/4$, and the failures geometric behind it has mean 3. With `!` the base mean is 4, $p = 1/5$, and the trials mean is 5. A zero-modified geometric is 0 with probability p_0^M and $1 + G$ otherwise, so its mean is $(1 - p_0^M)(1 + \text{base mean})$, here $(3/4)(1 + 13/3) = 4$.

```
geo = counts('geometric', 'geometric zt', 'geometric zt !', 'geometric zm 0.25')
GT(geo)
```

clause	P(N=0)	P(N=1)	mean	base mean	computed mean
geometric	0.2000	0.1600	4	4.0000	4.0000
geometric zt	0.0000	0.2500	4	3.0000	4.0000
geometric zt !	0.0000	0.2000	5	4.0000	5.0000
geometric zm 0.25	0.2500	0.1406	4	4.3333	4.0000

The distribution `scipy.stats.geom` is the trials form, and `geometric zt` matches it to within 10^{-17}. Shifting the failures geometric with mean 3 up by one gives the same law, as memorylessness requires.

```
k = np.arange(1, 50)
t = build('agg Trials 4 claims dsev [1] geometric zt',
          bs=1, log2=8).density_df.p_total
g = build('agg Failures 3 claims dsev [1] geometric',
          bs=1, log2=8).density_df.p_total
print(np.abs(t.loc[k].values - stats.geom(1/4).pmf(k)).max())
print(np.abs(t.loc[k].values - g.loc[k - 1].values).max())
```

```
7.15573433840433e-18
8.964996713739515e-18
```

Refused programs

A zero-truncated count cannot have a mean of one claim or fewer, since every outcome is at least one, and the error message gives the `!` form as an alternative. A count outside the (a, b, 1) class, such as `fixed`, has no zero modification. The value given to `zm` is a probability $\Pr(N = 0)$ that must lie in $[0, 1)$, since 1 leaves no probability for a claim.

```
refuse('agg X 0.5 claims dsev [1] poisson zt',
       'agg X 1 claim dsev [1] geometric zt',
       'agg X 4 claims dsev [1] fixed zt',
       'agg X 4 claims dsev [1] poisson zm 1',
       'agg X 4 claims dsev [1] poisson zm -0.2')
```

```
solve_base_mean: 'poisson' with p0 = 0 cannot reach a mean of 0.5; zero
modification admits only E[N] > 1. Append ! to read the count as the un-modified
base mean and let the modification shift it, or raise the claim count.
solve_base_mean: 'geometric' with p0 = 0 cannot reach a mean of 1; zero
modification admits only E[N] > 1. Append ! to read the count as the un-modified
base mean and let the modification shift it, or raise the claim count.
Zero modification not implemented for fixed
zm: P(N = 0) must lie in [0, 1), not 1.0
zm: P(N = 0) must lie in [0, 1), not -0.2
```