Using unconditional layers

Conditional and unconditional layer severities, and the trailing ! on a severity

Stephen J. Mildenhall

2026-11-02

A layer y xs a pays $g(X) = \min((X-a)^+, y)$ on a ground-up loss X. It has two natural severities. The **conditional** severity is the law of $g(X)$ given $X > a$, the payment on a loss that reaches the layer. The **unconditional** severity is the law of $g(X)$ itself, which includes a zero payment for each loss that does not reach it. The two are related by

$$\Pr(g(X) \le t) = \Pr(X \le a) + \Pr(X > a)\ \Pr(g(X) \le t \mid X > a), \quad t \ge 0,$$

so the unconditional severity is the conditional one with an extra mass $\Pr(X \le a)$ at zero, and $\mathsf{P}g(X) = \Pr(X > a)\,\mathsf{P}[g(X) \mid X > a]$. The conditioning is strict: a loss of exactly a pays nothing and is excluded.

In DecL the layers clause `y xs a` sits between the exposure and the severity. By default the severity is conditional, and the claim count in the exposure clause is the count of losses that reach the layer. A trailing `!` after the severity makes it unconditional: the claim count is the ground-up count, and each loss below the attachment adds a zero payment. It is used when the claim count is a ground-up count, from a ground-up frequency study or one claim per policy, and the payment to a layer is needed. A program with no layers clause uses the ground-up severity as it stands, including any mass at zero, and conditions on nothing. The grammar accepts `!` after any `sev` or `dsev`, but without a layers clause it has no effect. The same is true with a layers clause when $\Pr(X > a) = 1$.

```
agg Eg dfreq [1] dsev [0:4]             # ground up, no layers clause
agg Eg dfreq [1] dsev [0:4] !           # the same, nothing to condition on
agg Eg dfreq [1] 3 xs 0 dsev [0:4]      # min(X, 3) given X > 0
agg Eg dfreq [1] 3 xs 0 dsev [0:4] !    # min(X, 3)
agg Eg dfreq [1] 2 xs 2 dsev [0:4]      # min((X - 2)+, 2) given X > 2
agg Eg dfreq [1] 2 xs 2 dsev [0:4] !    # min((X - 2)+, 2)
```

```
import pandas as pd
from aggregate import build, qd, __version__
from style import wprint, GT, refuse, figures
figures()
print(__version__)
```

```
1.63.2
```

Conditional and unconditional densities of one claim

The six programs have one claim for sure, `dfreq [1]`, with a ground-up loss X equally likely to be 0, 1, 2, 3, or 4. With one claim the aggregate is the layer severity, so the column `p_total` of `density_df` is the density of the payment. The table shows the density under each program, one column per program, grouped by layers clause with the default form beside the form with `!`.

```
aggs = {}
for layer in ['', '3 xs 0', '2 xs 2']:
    for bang in ['', '!']:
        key = (layer or 'no layer', bang or 'default')
        aggs[key] = build(f'agg Eg dfreq [1] {layer} dsev [0:4] {bang}')
density = pd.DataFrame({k: a.density_df.p_total for k, a in aggs.items()})
GT(density.loc[0:4].fillna(0).rename_axis('payment'))
```

	no layer		3 xs 0		2 xs 2	
payment	**default**	**!**	**default**	**!**	**default**	**!**
0	0.2000	0.2000	0.0000	0.2000	0.0000	0.6000
1	0.2000	0.2000	0.2500	0.2000	0.5000	0.2000
2	0.2000	0.2000	0.2500	0.2000	0.5000	0.2000
3	0.2000	0.2000	0.5000	0.4000	0.0000	0.0000
4	0.2000	0.2000	0.0000	0.0000	0.0000	0.0000

With no layers clause the payment is the loss, with probability 1/5 on each outcome including zero, and `!` changes nothing. The layer 3 xs 0 pays $\min(X, 3)$. Conditioning on $X > 0$ removes the zero loss and scales the other four up by $1/\Pr(X > 0) = 5/4$, giving 1/4 at 1 and at 2, and 1/2 at the limit 3 from the losses of 3 and 4. With `!` the zero loss stays, and the losses of 3 and 4 put 2/5 at the limit. The layer 2 xs 2 pays nothing on losses of 0, 1, and 2. Conditioning on $X > 2$ leaves the losses of 3 and 4, which pay 1 and 2 with probability 1/2 each. With `!` the payment is zero with probability 3/5, the three losses at or below the attachment, and 1 or 2 with probability 1/5 each. In both layers the unconditional column is $\Pr(X \le a)$ at zero plus $\Pr(X > a)$ times the conditional column.

A layers clause with attachment zero is therefore different from no layers clause, because it conditions on $X > 0$. The layer `inf xs 0` keeps every loss but removes the zero, so it puts 1/4 on each of 1, 2, 3, and 4, with mean 5/2 in place of the ground-up mean 2.

```
a = build('agg Eg dfreq [1] inf xs 0 dsev [0:4]')
print(a.density_df.p_total.loc[0:4].values, a.est_m)
```

```
[0.   0.25 0.25 0.25 0.25] 2.5
```

Layer means and the probability of attaching

The attribute `pattach` of a severity is $\Pr(X > a)$, here 4/5 for 3 xs 0 and 2/5 for 2 xs 2. The conditional layer means are $(1 + 2 + 3 + 3)/4 = 9/4$ and $(1 + 2)/2 = 3/2$. Multiplying by `pattach` gives the unconditional means 9/5 and 3/5, the means of the `!` programs.

```
rows = {}
for layer in ['3 xs 0', '2 xs 2']:
    c, u = aggs[(layer, 'default')], aggs[(layer, '!')]
    p = c.sevs[0].pattach
    rows[layer] = [p, c.est_m, p * c.est_m, u.est_m]
GT(pd.DataFrame.from_dict(rows, orient='index', columns=[
    'pattach', 'conditional mean', 'pattach x conditional',
    'unconditional mean']).rename_axis('layer'))
```

layer	pattach	conditional mean	pattach x conditional	unconditional mean
3 xs 0	0.8000	2.2500	1.8000	1.8000
2 xs 2	0.4000	1.5000	0.6000	0.6000

Ground-up and layer claim counts

The `!` changes the meaning of the claim count in the exposure clause, from a count of losses to the layer to a count of ground-up losses. A count of ground-up losses can always be converted into a count of losses to the layer. Each ground-up loss reaches 2 xs 2 with probability 2/5. The single ground-up claim of `dfreq [1]` therefore gives a Bernoulli count to the layer with mean 2/5, and a Poisson count of 10 ground-up claims thins to a Poisson count of 4 claims to the layer. Each pair of programs below has the same aggregate distribution, the first unconditional and the second conditional; the maximum difference in `p_total` is at the level of rounding.

```
pairs = [('agg U dfreq [1] 2 xs 2 dsev [0:4] !',
          'agg C 0.4 claims 2 xs 2 dsev [0:4] bernoulli'),
         ('agg U 10 claims 2 xs 2 dsev [0:4] ! poisson',
          'agg C 4 claims 2 xs 2 dsev [0:4] poisson')]
for pu, pc in pairs:
    u, c = build(pu, bs=1, log2=8), build(pc, bs=1, log2=8)
    diff = (u.density_df.p_total - c.density_df.p_total).abs().max()
    print(f'mean {u.est_m:.4f} {c.est_m:.4f}   max difference {diff:.2e}')
```

```
mean 0.6000 0.6000   max difference 5.55e-17
mean 6.0000 6.0000   max difference 5.55e-17
```

The two forms differ in which count is declared. With `!` the exposure clause holds the ground-up count, and the count to the layer follows from the severity. A layer moved to a new attachment then keeps the same exposure clause, where the conditional form needs a new claim count for each attachment. The deductible forms of the layers clause are on Using deductibles. The `!` after `zt` or `zm` in the frequency clause is unrelated: it sets the base mean of a modified count, and is on Using `zm` and `zt`.