Using `pnl` and `xpnl`

Profit and loss statements with the pnl and xpnl kinds and the PnL class

Stephen J. Mildenhall

2026-10-30

An aggregate gives the distribution of loss, but an underwriting result also includes premium and expense, and, with reinsurance, ceded premium, recoveries, and ceding commission. The DecL kinds `pnl` and `xpnl` wrap a complete aggregate or portfolio, which models the loss, and return a `PnL`: the distribution of the underwriting result together with its accounting ledger.

With gross premium P, expense E, gross loss X, ceded premium C, recovery R, and ceding commission K, the net result is

$$M = (P - C) - (X - R) - (E - K).$$

The net result needs only the law of the net loss $X - R$, which the wrapped aggregate computes. A **walk** shows the gross result and the result of each cover separately and adds them outcome by outcome, so it needs the joint law of (X, R); for an occurrence cover, R depends on the individual claims and not only on X. A `PnL` uses the payoff convention: profit is positive, the adverse tail is the left one, and `P01`, the 1% quantile, is the 1 in 100 bad year.

The kinds `pnl` and `xpnl`[1] share one syntax and give two views of the business. A `pnl` is the net position, one group with net premium against net loss and expense. An `xpnl` is the walk, with one group per step: gross first, then the occurrence program, then the aggregate program, and finally `All`, the net.

```
pnl  NAME <premium> less <engine> [less <expenses>]
xpnl NAME <premium> less <engine> [less <expenses>] [peel top-down | peel bottom-up]

<premium>   10000 premium | inherit premium | derive premium      [as "label"]
<engine>    agg NAME <body> | agg.NAME | port NAME <units> | port.NAME
<expenses>  2000 fixed expenses | 0.25 premium expenses | 0.1 loss expenses
```

In the grammar, `<engine>` is the aggregate or portfolio that models the loss, written in full or as a reference to a stored entry. `inherit premium` uses the premium of that aggregate or portfolio, and `derive premium` grosses its premium P_0 up for the expenses, to $(P_0 + F)/(1 - r)$ for fixed expense F and premium expense ratio r. The separator is the keyword `less` rather than `-`, so it cannot be confused with severity arithmetic. In an `xpnl`, `peel` makes each layer of a program its own step, instead of one step per program.

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

```
1.43.1
```

[1] From aggregate 1.0

A ceding commission treaty as a pnl and an xpnl

The example is the entry `CedingCommission` in the recipe library included with aggregate (see Using recipes), shortened to `CC`. It is a book with premium 10,000 at a 70% loss ratio and policy limits of 2,000, one occurrence layer 1,750 xs 250 rated at 32.5% of gross premium with a 10% ceding commission, and 2,000 of fixed expense. The program is written once, with a placeholder for the kind, and built first as a `pnl`.

```
program = '''{kind} CC
  inherit premium
  less
    agg CC.Book
      10000 premium at 0.7 lr
      2000 xs 0
      sev lognorm 150 cv 2.5
      occurrence net of
        1750 xs 250 rate 0.325 cede 0.1 as "Occ Re"
      mixed gamma 0.25
  less
    2000 fixed expenses as Expenses'''
p = build(program.format(kind='pnl'))
qd(p)
```

```
PnL('CC': role=sell, 1 consideration, 3 obligation -> 'margin')

                 EX     SD       CV     Skew   P01  Median   P99
Side
Consideration  6750      0        0           6750    6750  6750
Obligation    -6156 1421.4 -0.23089 -0.52485 -9995   -6032 -3410
Margin          594 1421.4   2.3929 -0.52485 -3245     718  3340
```

The `pnl` shows the net position. Net premium 6,750 is the gross 10,000 less 3,250 ceded, and the obligation, 6,156, is net loss plus expense less commission. Built as an `xpnl`, the same program gives the walk: the gross step, the cover step `Occ Re`, then `All`. The cover step ends with the running `Net`, and `All` adds the `Impact`, the margin of the whole reinsurance program.

```
x = build(program.format(kind='xpnl'))
qd(x)
```

```
PnL('CC': 2 groups, 6 legs -> 'margin')

                        EX     SD       CV     Skew    P01  Median   P99
Step   Side
Gross  Consideration 10000      0        0           10000   10000 10000
       Obligation    -9000 2706.5 -0.30072 -0.67617 -16560   -8680 -4120
       Margin         1000 2706.5   2.7065 -0.67617  -6560    1320  5880
Occ Re Consideration -3250      0        0           -3250   -3250 -3250
       Obligation     2844 1619.5  0.56944  0.94221    405    2605  7645
       Margin         -406 1619.5  -3.9889  0.94221  -2845    -645  4395
       Net             594 1426.7   2.4019  -0.5251  -3245     715  3355
All    Consideration  6750      0        0            6750    6750  6750
       Obligation    -6156 1426.7 -0.23176  -0.5251  -9995   -6035 -3395
       Margin          594 1426.7   2.4019  -0.5251  -3245     715  3355
       Impact         -406 1619.5  -3.9889  0.94221  -2845    -645  4395
```

The margin of the cover is −406: the cedant pays 406 a year on average for the protection, and the standard deviation of the result falls from 2,706 to 1,427. The two kinds agree on the mean, 594. The walk is computed from the joint law of gross and ceded loss on a coarser grid, so its standard deviation, 1,427, differs from the 1,421 of the `pnl` in the fourth significant figure. The wrapped

aggregate is the `engine` attribute of a `PnL`, and its mean, 4,481, is the mean net loss. On the `pnl`, `q(0.01)` is the 1 in 100 result, a loss of 3,245, and `cdf(0)`, the probability that the result is at most zero, is 0.312.

```
print(p.est_m, x.est_m)
print(p.est_sd, x.est_sd)
print(p.q(0.01), p.cdf(0), x.engine.est_m)
```

```
593.9985626922165 593.9985628368244
1421.35256663098 1426.7030414293781
-3245.0 0.3122290184639707 4481.00143731014
```

With `derive premium` in place of `inherit premium`, the 2,000 of fixed expense is added to the 10,000 premium of the aggregate, for a gross premium of 12,000. The ceded premium, 32.5% of gross, rises with it to 3,900.

```
d = build(program.format(kind='xpnl').replace('inherit', 'derive'))
print(d.economic_ratios_df.P)
```

```
Step
Gross     12000.0
Occ Re    -3900.0
All        8100.0
Name: P, dtype: float64
```

The accounting ledger

The accounting ledger of a `PnL` is the table `economic_df`, which lists every leg by step and side. Column `κ01` is the conditional mean of each row given that the net result is at its 1% quantile, and likewise `κ50` and `κ99`. Each κ column is one scenario and foots down the table: in the 1 in 100 year the gross margin of −5,380 and the margin of the cover, 2,135, give the net −3,245.

```
GT(x.economic_df[['EX', 'SD', 'κ01', 'κ50', 'κ99']], float_format=',.0f')
```

Step	*Side*	*Label*	**EX**	**SD**	**κ01**	**κ50**	**κ99**
Gross	Consideration	Premium	10,000	0	10,000	10,000	10,000
	Obligation	Loss	-7,000	2,706	-13,380	-6,793	-2,469
		Expenses	-2,000	0	-2,000	-2,000	-2,000
		Total	-9,000	2,706	-15,380	-8,793	-4,469
	Margin	Gross	1,000	2,706	-5,380	1,207	5,531
Occ Re	Consideration	Occ Re premium	-3,250	0	-3,250	-3,250	-3,250
	Obligation	Occ Re recovery	2,519	1,619	5,060	2,433	749
		Occ Re commission	325	0	325	325	325
		Total	2,844	1,619	5,385	2,758	1,074
	Margin	Total	-406	1,619	2,135	-492	-2,176
		Net	594	1,427	-3,245	715	3,355
All	Consideration	Net	6,750	0	6,750	6,750	6,750
	Obligation	Net	-6,156	1,427	-9,995	-6,035	-3,395
	Margin	Net	594	1,427	-3,245	715	3,355
		Impact	-406	1,619	2,135	-492	-2,176

Amounts and ratios by step

The table `economic_ratios_df` has one row per step of the walk. It contains premium `P`, loss `L`, expense `E`, margin `M`, and the standard deviation `SD` of the result, signed so that the amounts add across steps. It also contains the loss, expense, and combined ratios `LR`, `ER`, and `CR`. On the cover row `E` is the ceding commission, -325, and on `All` the expense is net of it, $2{,}000 - 325 = 1{,}675$.

```
GT(x.economic_ratios_df[['P', 'L', 'E', 'M', 'SD', 'LR', 'ER', 'CR']],
   float_format=',.0f', ratio_cols=['LR', 'ER', 'CR'])
```

Step	P	L	E	M	SD	LR	ER	CR
Gross	10,000	7,000	2,000	1,000	2,706	70.0%	20.0%	90.0%
Occ Re	-3,250	-2,519	-325	-406	1,619	77.5%	10.0%	87.5%
All	6,750	4,481	1,675	594	1,427	66.4%	24.8%	91.2%

Margin and capital by step

The table `waterfall_df` sets the margin of each step against the capital behind it, at the return period stored in its `attrs`, here 100. The column `Capital standalone` is the negated 1 in 100 result of the step alone; for a cover it comes from the right tail of the cover's own result and is negative, capital released. `Capital net` conditions every step on the whole book at its 1 in 100 result, so it foots to the net capital, $5{,}380 - 2{,}135 = 3{,}245$. `MSD` is margin over its standard deviation, and `CoC net`, the cost of capital, is margin over net capital. On a `pnl` the table is empty, since a single group has no steps.

```
w = x.waterfall_df
print(w.attrs)
GT(w[['Margin', 'MSD', 'Capital standalone', 'Capital net', 'CoC net']],
   float_format=',.0f', formatters={'MSD': '{x:.3f}'}, ratio_cols=['CoC net'])
```

```
{'return_period': 100}
```

Step	Margin	MSD	Capital standalone	Capital net	CoC net
Gross	1,000	0.369	6,560	5,380	18.6%
Occ Re	-406	-0.251	-4,395	-2,135	19.0%
All	594	0.416	3,245	3,245	18.3%

The pentagon of each step at a return period

The **pentagon** of a step is the set of five amounts loss L, margin M, premium $P = L + M$, capital Q, and assets $a = P + Q$. The method `pentagon_df(t)` computes it for each step at return period `t`, with the technical premium `P_tech` $= L + M$ and the capital `Q` equal to the negated 1 in `t` result. It adds the cost of capital `CoC` $= M/Q$, the loss ratio `LR` on written premium, and the technical loss ratio `TLR` on technical premium. The `Ceded` row is the whole cession, with capital the difference between `All` and `Gross`, $3{,}925 - 7{,}960 = -4{,}035$ at a 250 year return period.

```
GT(x.pentagon_df(250)[['L', 'M', 'P_tech', 'Q', 'a', 'CoC', 'LR', 'TLR']],
   float_format=',.0f', ratio_cols=['CoC', 'LR', 'TLR'])
```

Step	L	M	P_tech	Q	a	CoC	LR	TLR
Gross	7,000	1,000	8,000	7,960	15,960	12.6%	70.0%	87.5%
Occ Re	-2,519	-406	-2,925	-5,315	-8,240	7.6%	77.5%	86.1%
All	4,481	594	5,075	3,925	9,000	15.1%	66.4%	88.3%
Ceded	-2,519	-406	-2,925	-4,035	-6,960	10.1%	77.5%	86.1%

Refused programs

Three P&L programs are refused: `inherit premium` on an aggregate with no premium, `peel` on a `pnl`, which has no steps, and `derive premium` with a loss expense, which the gross-up formula does not include. Each error message includes the fix.

```
bare = 'agg Bare 10 claims sev lognorm 10 cv 1 poisson'
priced = 'agg Priced 100 premium at 0.7 lr sev lognorm 10 cv 1 poisson'
refuse(f'pnl R1 inherit premium less {bare}',
       f'pnl R2 100 premium less {priced} peel top-down',
       f'pnl R3 derive premium less {priced} less 0.1 loss expenses')
```

```
R1: 'inherit premium' but the wrapped engine has no premium to inherit -- it
needs a 'premium at lr' exposure (or a stored agg / port carrying premium). Give
an explicit '<amount> premium' instead.
R2: 'peel' is an 'xpnl' clause. A 'pnl' is the consolidated single-group net
view, so it has no steps to peel; write 'xpnl R2 ... peel top-down' for the
layer-by-layer walk.
R3: 'derive premium' cannot gross up a loss basis expense; losses are not
reliably known by inspection. Use fixed or premium expenses, or 'inherit
premium' with the loss expense left in place.
```