

Bayesian Loss-Development Reserving with BLISS: A Vignette

Frank Schmid

This version: 4 July 2026

First version: 23 June 2026

Abstract. For many lines of insurance the ultimate loss associated with an exposure year is not known for many calendar years; the losses develop as time progresses. Loss development estimates the ultimate loss by exposure year, at the level of the aggregate loss triangle, from the losses observed to date. This vignette describes and demonstrates loss development with the package BLISS (Bayesian Loss-development Inference of Structural Segmentation), which centers on a Bayesian model of the development-year effect as a piecewise process in the rate of development, with the number and the location of the breakpoints inferred and model-averaged; a calendar-year effect identified by anchoring to an external macroeconomic level, with a reverting Ornstein–Uhlenbeck deviation about that anchor; exact enumeration of the development structure below a candidate-set cap, validated against a reversible-jump sampler above it; and a marginal-likelihood test of whether the calendar-year effect earns its place in the fit. The package is written in Python and depends only on NumPy and SciPy. The worked example is the State Farm Workers’ Compensation paid-loss triangle from the Meyers–Shi Schedule P loss-reserving database, the application of the methodology paper.¹ The package follows the precedent of the R package `lossDev`.²

¹Frank Schmid, “Identification and Bayesian Inference of Loss-Development Structure” (working paper, 2026).

²Christopher W. Laws and Frank A. Schmid, “Robust Loss Development Using MCMC: A Vignette,” 2010; Frank A. Schmid, “Robust Loss Development Using MCMC,” *Variance* 4, no. 2 (2010): 100–115.

Contents

Abbreviations and Acronyms	iii
1 Installation	1
2 Model Overview	2
3 The Worked Triangle	3
3.1 Data	3
3.2 Specification and Fitting	3
3.3 Examining the Fit	4
3.4 The Three Effects	6
3.5 The Reserve	10
3.6 The Tail	12
4 The Calendar-Year Effect: Does It Earn Its Place?	13
5 Structural Change in the Calendar-Year Effect	15
6 Scenarios, Reliability, and Deliverables	16
7 Reproducibility	18
Bibliography	19

Abbreviations and Acronyms

BF — Bayes factor

BLISS — Bayesian Loss-development Inference of Structural Segmentation

DY — development year

JAGS — Just Another Gibbs Sampler

MCMC — Markov chain Monte Carlo

OU — Ornstein–Uhlenbeck

RJMCMC — reversible-jump Markov chain Monte Carlo

USD — United States dollar

Greek Letters

β — beta, pronounced BAY-tuh; the development-year effect.

γ — gamma, pronounced GAM-uh; the calendar-year effect.

ε — epsilon, pronounced EP-sih-lon; the observation error.

σ — sigma, pronounced SIG-muh; the observation scale; σ_r , the development-rate scale; σ_C , the calendar-break magnitude scale; $\sigma_a(h)$, the anchor forecast-error standard deviation by horizon.

φ — phi, pronounced FY; the persistence of the Ornstein–Uhlenbeck deviation.

1 Installation

BLISS is a Python package distributed as a standalone archive, `bliss-0.8.0.zip`. Unpacking the archive and installing with `pip` makes the package importable; the core depends on NumPy, SciPy, pandas, and openpyxl, and the figure modules use matplotlib, installed through the `figures` extra. Unlike `lossDev`, which calls JAGS (Just Another Gibbs Sampler) through `rjags` for its Markov chain Monte Carlo (MCMC), BLISS carries its own samplers and its exact-enumeration path, so no external Monte Carlo engine is installed.

```
pip install .                # from the unpacked archive root
pip install "[figures]"      # with the optional figure modules
```

```
import bliss
bliss.__version__
```

```
'0.8.0'
```

The paper's worked-example data ship with the package under `bliss/data`. The replication archive of the accompanying paper, `Loss_Development_Structure_Replication.zip`, is maintained separately and is frozen to the paper; this vignette walks through the package workflow function by function, reproducing the paper's results.

2 Model Overview

BLISS identifies three time dimensions in the incremental payments of a loss triangle: the accident-year level, the development-year effect, and the calendar-year effect. The expected incremental payment of accident year i at development lag j is multiplicative in the three effects, $m_{ij} = \exp(L_i + \beta_j + \gamma_{i+j})$, where L_i is the accident-year level, β_j the development-year effect, and γ_t the calendar-year effect indexed by the diagonal $t = i + j$. The observation is placed on the cumulative amount: the observed cumulative paid loss C_{ij} is log-normal about the running sum of the incremental means,

$$\log C_{ij} = \log \left(\sum_{k=0}^j m_{ik} \right) + \varepsilon_{ij}, \quad (1)$$

with ε_{ij} a heteroskedastic Gaussian disturbance whose base scale is σ and whose development-year variance shape is estimated on the increments and accumulated to the cumulative cell.

The development-year effect. The development-year effect is modeled through its rate of change. The sequence of incremental development rates is piecewise constant, the pieces separated by breakpoints. The number of breakpoints and their locations are not fixed in advance; BLISS enumerates the admissible segmentations, scores each by its Laplace marginal evidence under a prior on the breakpoint count and on the breakpoint locations, and reports the posterior over segmentations. Every downstream quantity—the development-year effect, the reserve, the predictive distribution—is averaged over that posterior rather than conditioned on a single segmentation.

The calendar-year effect. A loss triangle does not identify the calendar-year effect from the accident-year and development-year effects without a restriction, because the three indices satisfy the age–period–cohort identity $t = i + j$. BLISS supplies the restriction by anchoring the calendar-year rate of growth to an external macroeconomic level—here the nominal gross domestic product of the United States from the International Monetary Fund World Economic Outlook—and modeling the within-sample departure from that anchor as a mean-reverting Ornstein–Uhlenbeck deviation. The anchor fixes the long-term trend and the macroeconomic variation; the reverting deviation carries the triangle-specific short-term movement.

Inference. Below a candidate-set cap the development segmentations are enumerated exactly, and the segmentation posterior is the normalized vector of evidence-times-prior weights. Above the cap a reversible-jump Markov chain Monte Carlo sampler walks the segmentation space. On a triangle small enough to enumerate, the sampler is validated against the exact posterior; the agreement is the package’s primary redundancy mechanism, and Section 3.2 shows it on the worked triangle.

3 The Worked Triangle

3.1 Data

The worked example is the State Farm Workers' Compensation paid-loss triangle, group code 1767 in the Meyers–Shi Schedule P loss-reserving database, accident years 1988 through 1997 by development lags zero through nine, in thousands of United States dollars. The triangle and the anchor are read with `read_triangle` and `read_anchor`, and the calendar backbone is built from the anchor with `anchor.backbone`. `read_triangle` accepts either a wide accident-year by development-lag matrix or a long, tidy table of one row per observed cell. Both files ship with the package under `bliss/data`: `bliss.examples.data_path` returns their installed locations, and `bliss.examples.paper_triangle` and `bliss.examples.paper_anchor` load them directly.

```
import bliss
from bliss.examples import data_path
tri = bliss.read_triangle(data_path("wc_tri.csv"))      # AY-by-lag matrix
anc = bliss.read_anchor(data_path("ol_weo_anchor.csv")) # annual levels, A/
  F status
bb = anc.backbone(tri, forecast_error_sd=bliss.weo_forecast_error_sd())
tri.ny, tri.years[0], tri.years[-1], round(bb.slope, 6)
```

```
(10, 1988, 1997, 0.054891)
```

The anchor carries annual levels of nominal gross domestic product with each year flagged actual or forecast. `backbone` differences the log levels to growth, anchors the in-sample calendar level at zero in 1988, takes the slope as the per-year increment over the diagonals the triangle spans, projects the forward path from the forecast levels, and extends it beyond the last forecast year at the terminal growth rate. The optional `forecast_error_sd` carries the World Economic Outlook forecast-error standard deviation by horizon, which is specific to that anchor; an anchor without such an estimate is built without it.

3.2 Specification and Fitting

The model is specified as a breakpoint model on the development-year effect with the Ornstein–Uhlenbeck calendar deviation about the anchored backbone. `fit_insample` builds the model context from the backbone and the deviation parameters—the development-rate scale, the persistence $\varphi = 0.45$, and the stationary standard deviation $\sqrt{V} = 0.0212$ —refines the observation scale by empirical Bayes, and enumerates the development segmentations, returning the fitted records and the context. The three constants carry the paper's estimates: the development-rate scale $\sigma_r = 0.5782$ is the root-mean-square of the triangle's saturated free-rate development slopes, and the persistence $\varphi = 0.45$ and the stationary standard deviation $\sqrt{V} = 0.0212$ are estimated in the paper, where the reversion rate is tested for identification against the calendar curvature the triangle resolves.³ On a new triangle the rate scale is re-centered on that triangle's own free-rate slopes, the amplitude $\sqrt{V}$ is supplied as a calibrated scale, and the reversion rate is tested for identification at that amplitude with the packaged test of Section 6.

```
spec = bliss.make_spec("breakpoint", "fixed")
```

³Schmid, *Loss-Development Structure*.

```
ctx, recs, v_inc, vcell = bliss.fit_insample(
    tri.cum_dict(), bb.slope, bb.level, spec=spec,
    SIG_R=0.5782, ou_phi=0.45, ou_V=0.0212**2)
sigma, diag = bliss.report(recs, ctx, "no-break")
```

```
[no-break] base scale sigma = 0.0247
DY2 inclusion = 1.000, beyond-DY2 max = 0.117
leading segmentations: [(2, 0.682), (2, 7), 0.091), (2, 6), 0.063)]
early (1988-1992) average = -0.665
```

The base observation scale is 0.0247 on the logarithmic scale. The development effect carries a single breakpoint at development lag two: that breakpoint enters every retained segmentation, with inclusion probability one, while no later development lag is included with probability above 0.117. The leading segmentation, a single break at lag two, carries posterior weight 0.682.

3.3 Examining the Fit

The breakpoint-count posterior. The posterior over the number of breakpoints is read with `posterior_over_K`. The package reports it from the exact enumeration and, as the redundancy check, from the collapsed reversible-jump sampler; the two agree to within half a unit in the third decimal place.

```
P_K, inclusion = bliss.posterior_over_K(recs, tri.ny)
```

Table 1. Posterior distribution of the number of breakpoints in the development-year effect, from the exact enumeration of the development segmentations and from the collapsed reversible-jump Markov chain Monte Carlo sampler. The two paths agree to within half a unit in the third decimal place, the package's primary redundancy check on the worked triangle. Source: `bliss.posterior_over_K` on the State Farm Workers' Compensation triangle.

Number of breakpoints K	Exact enumeration	Collapsed RJMCMC
1	0.682	0.682
2	0.277	0.278
3	0.037	0.037
4	0.004	0.003

Figure 1 plots the breakpoint-count posterior.

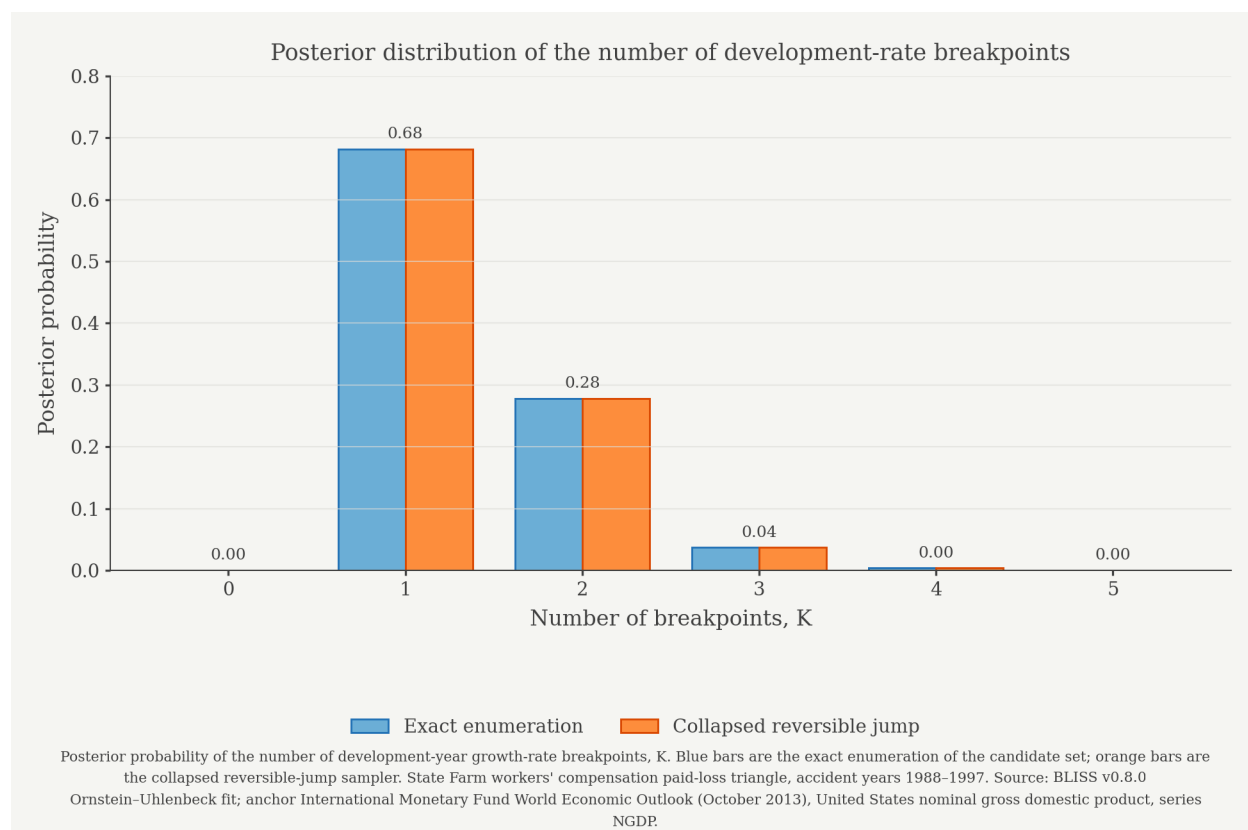


Figure 1. Posterior distribution of the number of breakpoints in the development-year effect. The horizontal axis is the number of breakpoints; the vertical axis is the posterior probability. Blue bars are the exact posterior from enumeration of all 64 segmentations with Laplace within-model evidence; orange bars are the collapsed reversible-jump estimate from four pooled runs. Source: `bliss.posterior_over_K` on the State Farm Workers' Compensation triangle.

Goodness of fit. The fit is summarized by its standardized residuals. The triangle holds 55 observed cells; the model-averaged base scale is 0.0247, and the standardized residuals have mean 0.006 and standard deviation 0.992, with a minimum of -2.65 and a maximum of 2.55 . Figure 2 plots the model-averaged fitted against the observed cumulative paid losses, one panel per accident year.

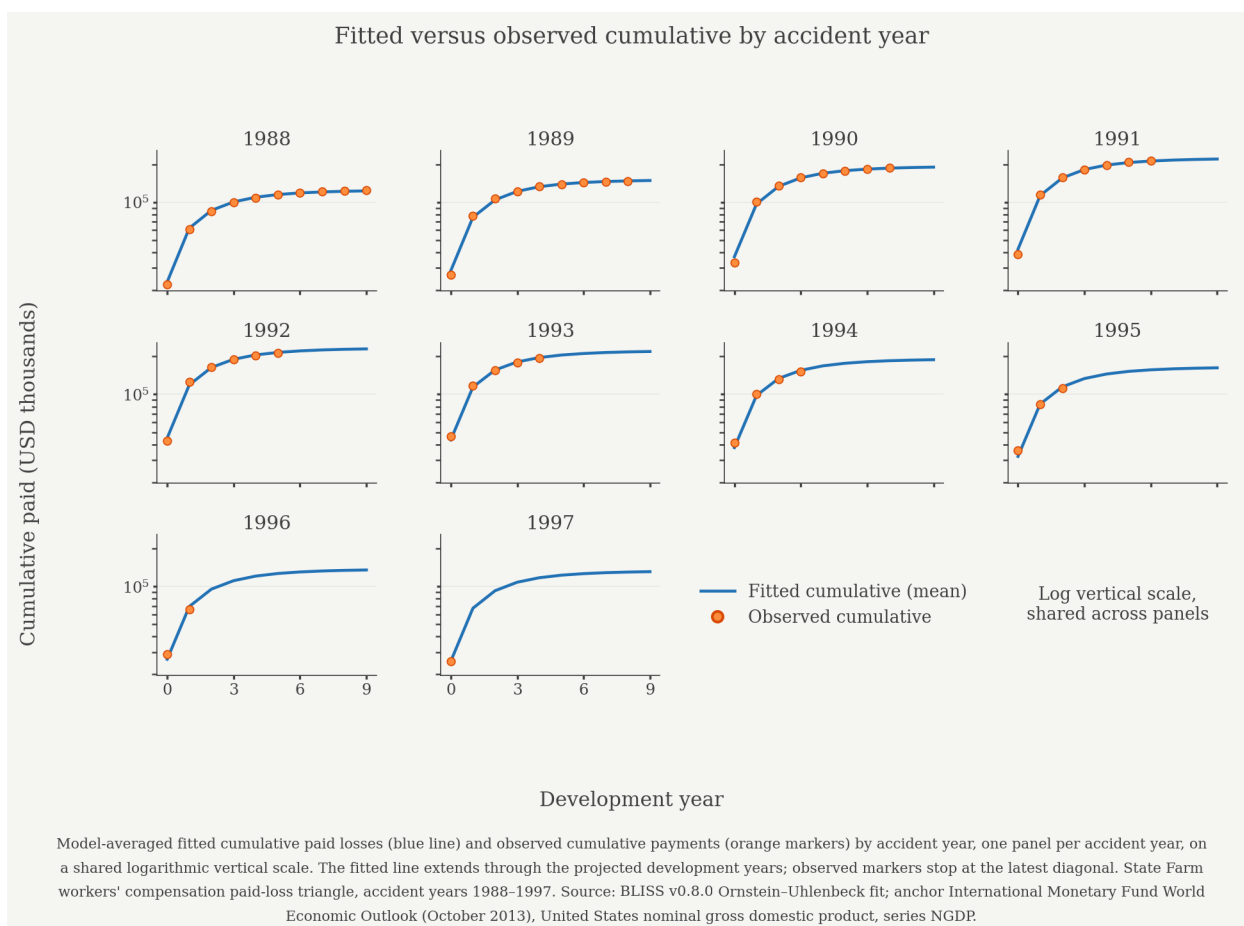


Figure 2. Model-averaged fitted against observed cumulative paid losses by accident year, one panel per accident year. The horizontal axis is the development year, zero through nine; the vertical axis is the cumulative paid loss in thousands of United States dollars on a logarithmic scale, shared across panels. The blue line is the model-averaged fitted cumulative mean; the orange markers are the observed cumulative payments. The bottom row holds the 1996 and 1997 panels in its first two cells; the third cell carries the legend and reading guide, and the fourth is empty. Source: the fitted records on the State Farm Workers' Compensation triangle.

3.4 The Three Effects

The accident-year level, the development-year effect, and the calendar-year effect are read from the fitted records with `aggregate_effects`, which returns each effect's posterior mean and a marginal credible band, model-averaged over the development segmentation.

```
eff = bliss.aggregate_effects(recs, ctx)
eff["beta"], eff["L"], eff["gamma"]    # development, accident-year level,
calendar
```

The development-year effect (Figure 3) is the cumulative profile of the incremental development rates. The single breakpoint at lag two separates a segment of rising incremental payments—a posterior-mean rate of +0.437 per lag before the break—from the long decay that follows, at −0.549 per lag.

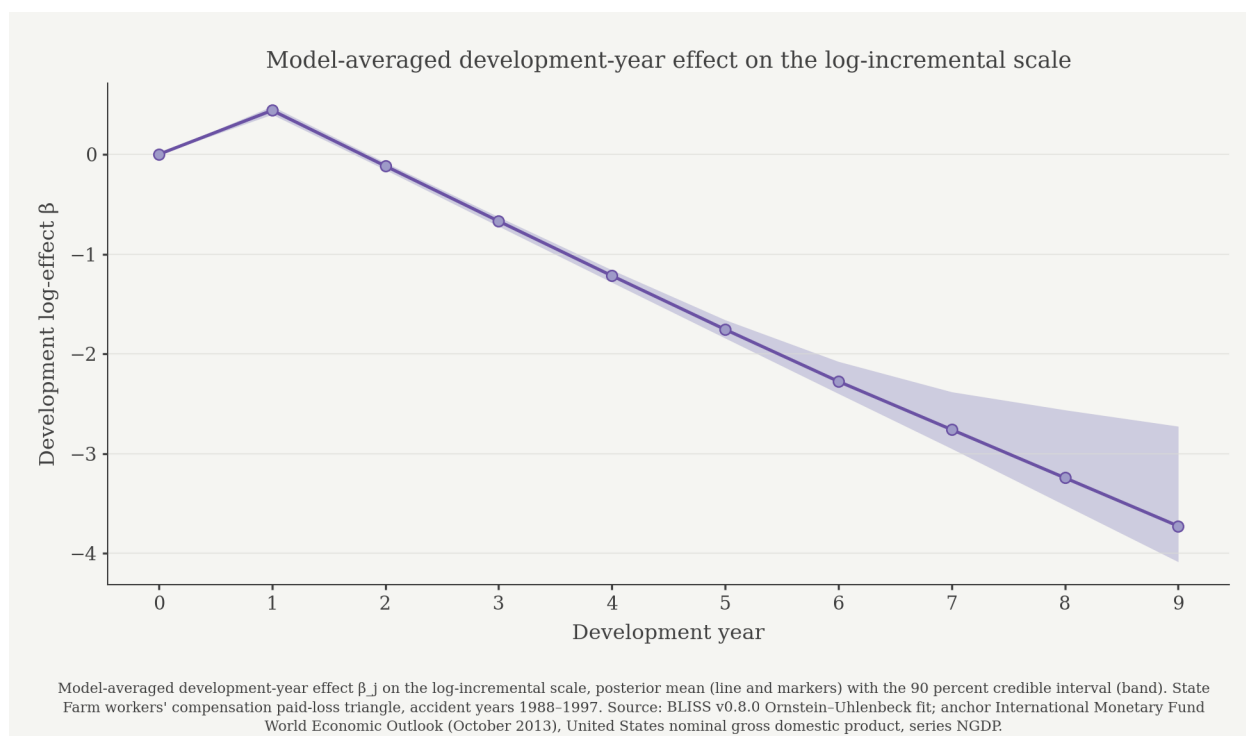


Figure 3. Model-averaged development log-effect (development year zero set to zero), the cumulative sum of the per-year growth rates. The line is the posterior mean and the band the 90 percent credible interval, both averaged over the breakpoint posterior. The horizontal axis is the development year; the vertical axis is the development log-effect. Source: `bliss.aggregate_effects` on the State Farm Workers' Compensation triangle.

The accident-year effect is shown through its year-on-year growth rate (Figure 4), the increment of the first-order random walk on the cohort levels.

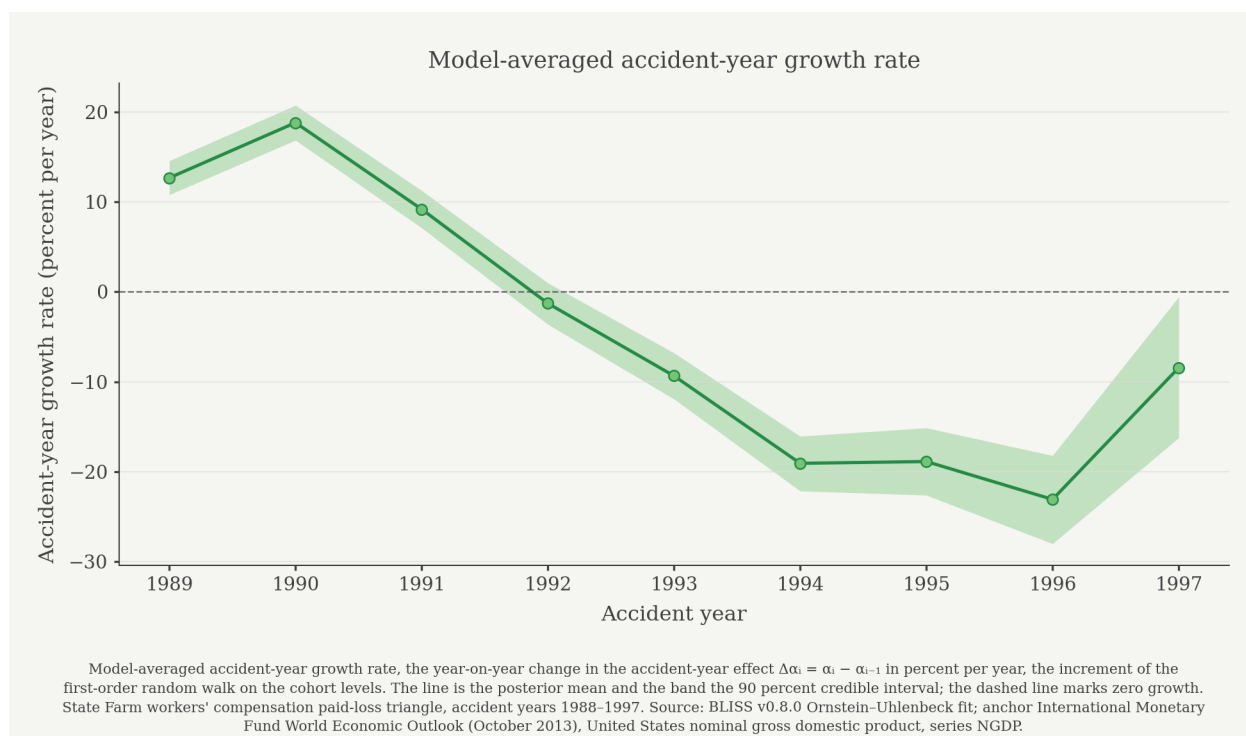


Figure 4. Model-averaged accident-year growth rate, the year-on-year change in the accident-year effect in percent per year, the increment of the first-order random walk on the cohort levels. The line is the posterior mean and the band the 90 percent credible interval; the dashed line marks zero growth. The horizontal axis is the accident year; the vertical axis is the growth rate in percent per year. Source: `bliss.aggregate_effects` on the State Farm Workers' Compensation triangle.

The calendar-year effect is the diagonal trend: the anchored backbone with its reverting Ornstein–Uhlenbeck deviation on the cumulative scale (Figure 5), and the same effect as a year-on-year growth rate (Figure 6).

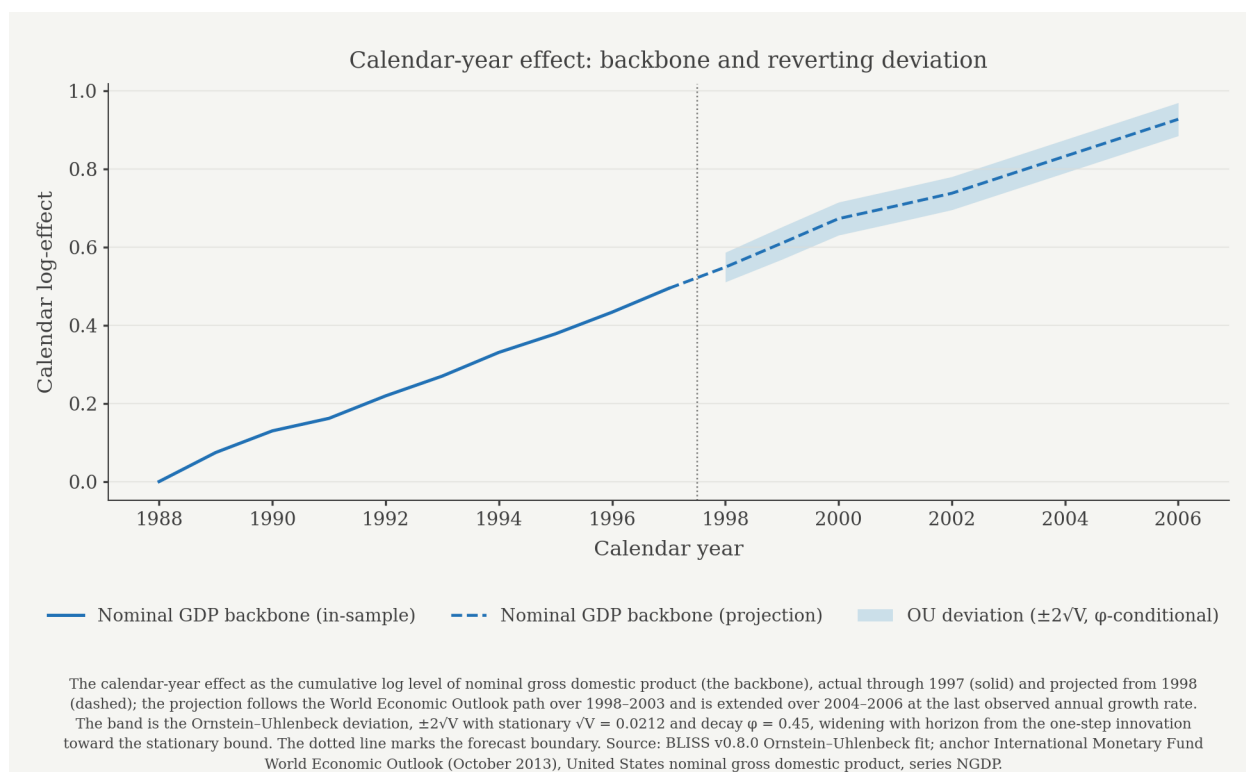


Figure 5. The calendar-year effect on the cumulative scale: the anchored backbone in sample, the projected backbone forward, and the bounded Ornstein–Uhlenbeck deviation band. The horizontal axis is the calendar year, 1988 through 2006; the vertical axis is the calendar-year effect on the log scale, the cumulative log calendar level relative to 1988. The solid line is the backbone, the cumulative log nominal gross domestic product, actual through 1997 and projected from 1998; the shaded band is the forward Ornstein–Uhlenbeck deviation. Source: the anchor series `ol_weo_anchor.csv`, International Monetary Fund World Economic Outlook, October 2013.

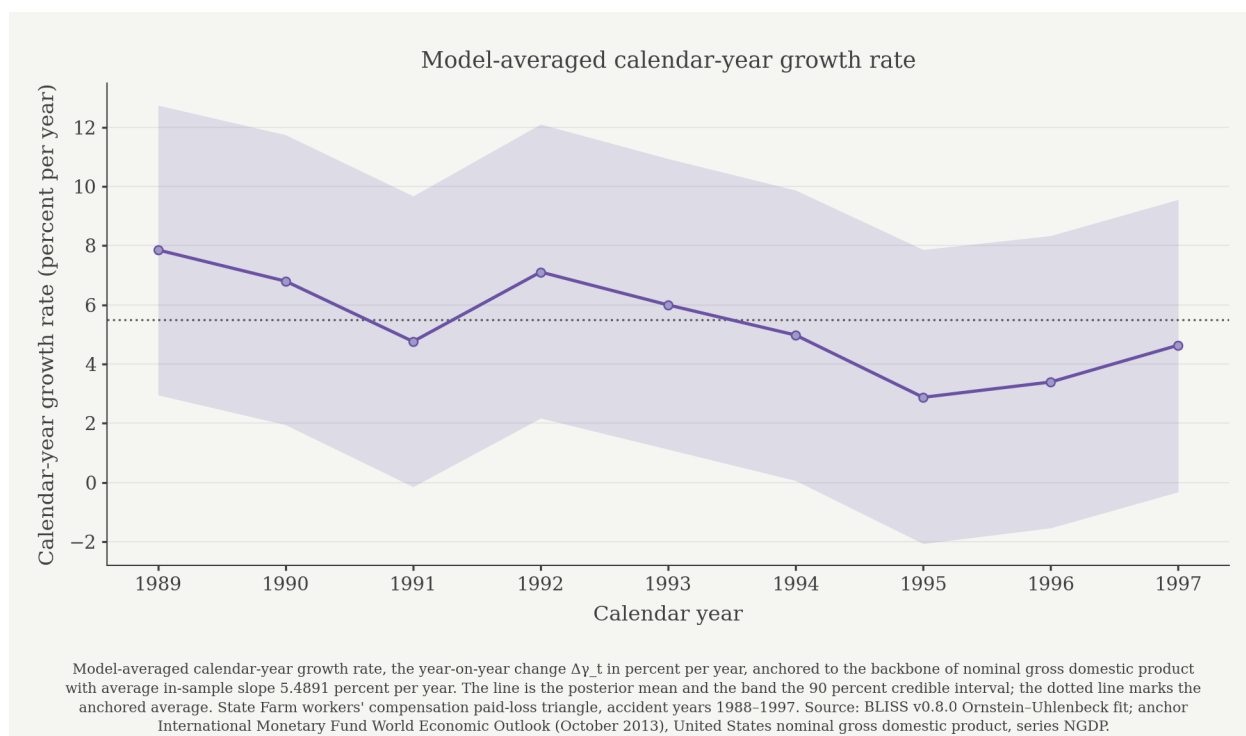


Figure 6. Model-averaged calendar-year growth rate, the year-on-year change in the calendar-year effect in percent per year, with the in-sample average anchored to the backbone of nominal gross domestic product at an in-sample slope of 5.4891 percent per year. The line is the posterior mean and the band the 90 percent credible interval; the dotted line marks the anchored average. The horizontal axis is the calendar year; the vertical axis is the growth rate in percent per year. Source: `bliss.aggregate_effects` on the State Farm Workers' Compensation triangle.

3.5 The Reserve

The outstanding loss is read with `outstanding_loss`, which takes the fitted records, the context, the backbone as the forecast object, and the paid-to-date, and returns the analytic log-normal mean, the predictive distribution from a draw over the posterior, the per-accident-year means, and the projected ultimate.

```
paid = tri.paid_to_date()
est = bliss.outstanding_loss(recs, ctx, bb, paid)
est.analytic_mean, est.predictive.median, est.predictive.geo_sd
```

| (313239.6, 300968.3, 1.0618)

The analytic mean of the total outstanding loss is USD 313,239.6 thousand, on a paid-to-date of USD 1,434,790 thousand, for a projected ultimate of USD 1,748,029.6 thousand. The predictive distribution of the outstanding loss (Figure 7) has a median of USD 300,968.3 thousand and a geometric standard deviation of 1.0618; its fifth and ninety-fifth percentiles are USD 278,648.4 thousand and USD 339,640.6 thousand; Figure 7 shows its posterior density. Table 2 reports the projection by accident year.

Table 2. Ultimate projection by accident year, model-averaged over the development segmentation. The latest paid is the most recent observed cumulative paid loss; the outstanding loss is the analytic mean of the future incremental payments; the projected ultimate is their sum. The 1988 accident year is fully developed and carries no outstanding loss. Amounts are in thousands of United States dollars. Source: bliss.outstanding_loss on the State Farm Workers' Compensation triangle.

Accident year	Latest paid	Outstanding loss	Projected ultimate
1988	125,049	0	125,049
1989	147,358	1,260.1	148,618.1
1990	187,760	3,923.6	191,683.6
1991	213,396	8,514.0	221,910.0
1992	213,904	15,083.3	228,987.3
1993	193,676	23,964.5	217,640.5
1994	151,081	34,370.3	185,451.3
1995	111,268	49,296.3	160,564.3
1996	66,033	67,940.1	133,973.1
1997	25,265	108,887.3	134,152.3
Total	1,434,790	313,239.6	1,748,029.6

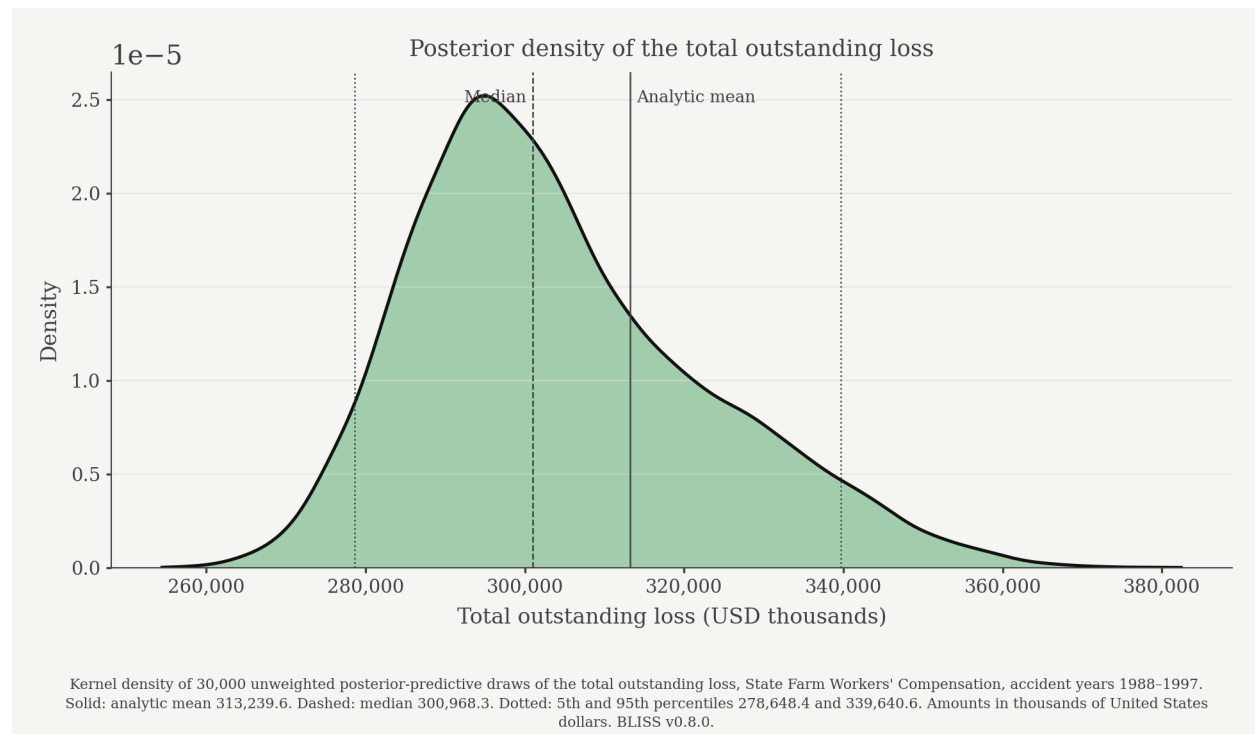


Figure 7. Posterior density of the total outstanding loss, integrating over the breakpoint structure, the segment rates, the accident-year levels, the calendar-year curvature, and the Ornstein–Uhlenbeck calendar forecast, a kernel estimate of 30,000 unweighted draws. The solid line marks the analytic mean, the dashed line the median, and the dotted lines the 5th and 95th percentiles. The horizontal axis is the total outstanding loss in thousands of United States dollars; the vertical axis is the posterior density. Source: bliss.outstanding_loss on the State Farm Workers' Compensation triangle.

3.6 The Tail

The tail is the set of development years past the last observed development year. Its length is a user choice through the `tail` argument of `outstanding_loss`, with a default of zero; this vignette uses five. In the tail the development effect continues at the final segment's growth rate—the year-on-year change of the development effect is piecewise constant, so the final segment's rate is the model's own continuation—and the calendar path continues along the forward anchor path, extended at the terminal forecast growth rate. The backbone is built with the same `tail` so the forward calendar path covers the tail diagonals; a forward path too short for the requested tail raises an error naming the remedy. The tail factor is the ratio of the expected ultimate loss including the tail to the expected ultimate loss excluding the tail. The outstanding loss, the predictive distribution, and the projected ultimate include the tail payments.

```
bb5 = anc.backbone(tri, tail=5,
                   forecast_error_sd=bliss.weo_forecast_error_sd())
est5 = bliss.outstanding_loss(recs, ctx, bb5, paid, tail=5)
est5.tail_factor, est5.tail_mean, est5.analytic_mean
```

```
(1.021402, 37411.6, 350651.2)
```

With five development years in the tail, the expected tail payments are USD 37,411.6 thousand on an expected ultimate excluding the tail of USD 1,748,029.6 thousand—the paid-to-date of USD 1,434,790 thousand plus the pre-tail outstanding loss of USD 313,239.6 thousand—for a tail factor of 1.021402. The outstanding loss including the tail is USD 350,651.2 thousand, and the projected ultimate including the tail payments is USD 1,785,441.2 thousand. The predictive distribution of the outstanding loss including the tail has a median of USD 316,435.4 thousand and a geometric standard deviation of 1.1320, against 1.0618 without the tail: each tail year loads the final segment's growth rate once more, so that rate's posterior spread compounds with the tail length, and the widening falls on the right tail of the distribution.

4 The Calendar-Year Effect: Does It Earn Its Place?

BLISS provides two readings of whether the calendar-year effect earns its place in the fit. The descriptive reading, `calendar_value`, refits three nested calendar specifications—the full anchored backbone with the reverting deviation, the flat anchor with the deviation retained, and no calendar-year effect at all—and reports each one’s base scale and the residual arc over the first diagonals. The inferential reading, supplied through the `evidence_fn` argument, adds the marginal-likelihood Bayes factor between the specifications, holding the observation noise shape common so the ratio isolates the calendar mean structure and penalizes the Ornstein–Uhlenbeck curvature through the Laplace determinant.

The three model contexts are constructed by builders over the noise shape; the common-shape evidence function is supplied by `make_evidence_fn`.

```
import numpy as np
cum = tri.cum_dict()
flat = bb.level - bb.slope * np.arange(tri.ny)
builders = {
    "full": lambda vdev=None, vcell=None: bliss.build_context(
        cum, bb.slope, SIG_R=0.5782, SIG_G=1.0, TAU_SCALE=0.5, ncurv=None,
        calendar="ou", vdev=vdev, vcell=vcell, backbone=bb.level,
        ou_phi=0.45, ou_V=0.0212**2),
    "flat_anchor": lambda vdev=None, vcell=None: bliss.build_context(
        cum, 0.0, SIG_R=0.5782, SIG_G=1.0, TAU_SCALE=0.5, ncurv=None,
        calendar="ou", vdev=vdev, vcell=vcell, backbone=flat,
        ou_phi=0.45, ou_V=0.0212**2),
    "no_calendar": lambda vdev=None, vcell=None: bliss.build_context(
        cum, 0.0, SIG_R=0.5782, SIG_G=1.0, TAU_SCALE=0.5, ncurv=0,
        calendar="ridge", vdev=vdev, vcell=vcell, backbone=None),
}
fit_eb = lambda cb: bliss.fit_insample(cum, bb.slope, bb.level,
                                       spec=spec, ctx_builder=cb)
cv = bliss.calendar_value(fit_eb, builders,
                          evidence_fn=bliss.make_evidence_fn(spec))
print(cv.summary())
```

```
Calendar-year value (earns-its-keep test; arc over the first 5 diagonals)
full      (anchored backbone + reverting OU deviation): sigma = 0.0247, early-arc = -0.665
flat_anchor (slope set to zero, deviation retained):      sigma = 0.0247 (-0.1%), early-arc =
-0.672
no_calendar (calendar-year effect constrained to zero):  sigma = 0.0281 (+13.8%), early-arc =
-0.788
verdict: the calendar-year axis earns its place in the fit through its curvature
        (no-calendar base scale rises +13.8%); the slope is fit-neutral
        (flat-anchor base scale identical to the full fit).

Calendar-year evidence (natural-log Bayes factor; observation noise shape
held common; bracket spans the full-fit and no-calendar-fit noise shapes)
axis (full vs no-calendar) log BF = +0.75 to +0.75 (not worth more than a bare mention for
the calendar-year axis)
curvature (flat vs no-cal.) log BF = +1.11 to +1.11 (positive for the reverting deviation)
slope (full vs flat anchor) log BF = -0.36 to -0.36 (not worth more than a bare mention for
the flat anchor)
```

Removing the calendar-year effect raises the base scale by 13.8 percent, from 0.0247 to 0.0281,

and reinstates the early-1990s residual arc, from -0.665 to -0.788 ; the reverting deviation carries that improvement. The flat anchor leaves the base scale unchanged, the slope re-absorbed by the accident-year and development-year effects. The Bayes factor charges the in-sample gain against the parameter cost: the curvature earns positive evidence, at a Bayes factor of 3.0; the slope's evidence is negative, the age–period–cohort-aliased slope carrying a parameter cost without a fit gain; and the whole axis is not worth more than a bare mention, the curvature's positive evidence partly offset by the slope's cost.

5 Structural Change in the Calendar-Year Effect

Where the consumption path is suspected of a structural break in a particular calendar year, BLISS fits a calendar-year breakpoint jointly with the development segmentation. The function `infer_calendar_break` enumerates the calendar breakpoint location together with the 64 development segmentations and returns the posterior probability of a break and its most probable location, under a prior on the magnitude of the break governed by the scale σ_C . This is the analog of the change-point model of `lossDev`.

```
builder = lambda: builders["full"](vdev=v_inc, vcell=vcell)
jres = bliss.infer_calendar_break(spec, builder, bliss.Provenance(),
                                phi=0.45, V=0.0212**2, sig_C=0.05)
top = max((c for c in jres["configs"] if c["K"] > 0),
          key=lambda c: c["Pcfg"])
print("P(break) = %.3f, location = %d" % (jres["Pbreak"], 1988 + top["tk"]))
```

```
| P(break) = 0.798, location = 1993
```

At a magnitude scale of $\sigma_C = 0.05$ the joint enumeration places posterior probability 0.798 on a calendar-year break, most probably at 1993, against 0.202 on no break. The posterior probability of a break rises with the magnitude scale, from 0.492 at $\sigma_C = 0.03$ to 0.979 at $\sigma_C = 0.15$, the location holding at 1993 throughout.

6 Scenarios, Reliability, and Deliverables

The workflow of Section 3 is carried end to end by `fit_ou`, which builds the backbone from the anchor, runs the in-sample fit, and returns a `Fit` object whose methods deliver the estimate, the reliability reading, the workbook, and the figures. The reliability reading classifies the fit from intrinsic signals—a reserve that pivots off the analytic mean, posterior mass on regularized configurations, or an analytic mean far above the predictive median all flag a weakly identified triangle.

```
f = bliss.fit_ou(tri, anc, SIG_R=0.5782, ou_phi=0.45, ou_V=0.0212**2)
est = f.estimate()
f.reliability(est)
```

```
| ('reliable', [])
```

`Fit.workbook` writes the fit's results to an Excel workbook whose Log tab records the author and the creation date, and `Fit.figures` writes the standard figure set, each figure verified at its native print width. Both operate on the anchored Ornstein–Uhlenbeck fit as returned by `fit_ou`.

The sensitivity of the reserve to the forward calendar path is read with `shock_scenarios`, which shocks the growth rate of the first forecast diagonal and decays the shock forward at the deviation's persistence.

```
for r in bliss.shock_scenarios(recs, ctx, bb, paid):
    print("%+d pp: outstanding %.1f" % (r["shock_pp"], r["outstanding"]))
```

```
-2 pp: outstanding 304792.4
-1 pp: outstanding 308985.5
+0 pp: outstanding 313239.6
+1 pp: outstanding 317555.4
+2 pp: outstanding 321934.0
```

A one-percentage-point shock to the first forward growth rate raises the reserve by USD 4,315.8 thousand and a negative one-point shock lowers it by USD 4,254.1 thousand, each about 1.4 percent of the reserve. Where specification uncertainty is material, `OutstandingLoss.calibrated` recenters the predictive distribution and inflates its log-dispersion in quadrature with a specification standard deviation estimated from a portfolio backtest; the portfolio calibration is the subject of a separate paper, and this vignette carries no value for it.

The deviation parameters enter the fit as given (Section 3.2); an optional diagnostic, `profile_ou`, maps the marginal evidence over a $(\varphi, \sqrt{V})$ grid at the standing fit's observation-noise shape. Holding the noise shape common makes the evidence comparable across grid points, the construction of Section 4. On this triangle the surface rises monotonically in the amplitude $\sqrt{V}$ at every persistence, so the surface carries no amplitude point estimate; within the fixed-shape construction the reversion rate is the resolved direction, read conditionally on the amplitude, and the amplitude itself is profiled by the empirical-Bayes construction below. Per amplitude, `profile_ou` reports twice the natural-log Bayes factor of the reverting family ($\varphi > 0$, uniform over the grid) against the independent-innovations boundary ($\varphi = 0$, the form the deviation reduces to where the data do not identify the reversion rate), graded on the Kass and Raftery scale,⁴ with the peak interior per-

⁴Robert E. Kass and Adrian E. Raftery, "Bayes Factors," *Journal of the American Statistical Association* 90, no. 430 (1995): 773–795.

sistence alongside. Per-point results checkpoint to a file, so an interrupted profile resumes rather than recomputes; the checkpoint carries the package’s core-defaults fingerprint and refuses to resume across a defaults change. The single-amplitude form of the test is `identify_ou_reversion`, which returns the reading and the point estimate at a supplied amplitude directly.

```
prof = bliss.profile_ou(cum, bb.slope, bb.level, SIG_R=0.5782,
                        vdev=v_inc, vcell=vcell,
                        checkpoint="ou_profile.pkl")
c = prof["conditional"][0.0212]
print("2 log BF = %.2f (%s); peak phi = %.2f (peak margin %.2f)"
      % (c["two_log_bf"], c["grade"], c["peak_phi"], c["peak_two_log_bf"]))
```

```
| 2 log BF = 1.27 (not worth more than a bare mention); peak phi = 0.60 (peak margin 1.95)
```

At the standing amplitude $\sqrt{V} = 0.0212$ the persistence profile peaks in the interior, at $\varphi = 0.60$ with a peak-against-boundary margin of 1.95, and the family reading of 1.27 sits below the Kass–Raftery positive grade; the family reading strengthens with the amplitude, reaching 2.61 at $\sqrt{V} = 0.030$. The diagnostic reads the evidence at a fixed noise shape and under the family construction stated above; the paper’s identification test is a separate construction, its finding is carried in Section 3.2 as given, and the paper’s Appendix C reports the marginal-likelihood audit of both deviation parameters.

The amplitude, supplied as a scale in the fit, is profiled by the empirical-Bayes marginal likelihood with `profile_ou_amplitude`: at each grid amplitude the full in-sample empirical-Bayes loop re-runs, the observation-noise shape re-equilibrates, and the point is read by the total marginal likelihood at its converged shape. The evidence carries the noise-shape determinant term, and the pipeline normalizes each shape to geometric mean one over the observed cells, so the per-point readings are comparable; the profile asserts that invariant at every point, and per-point results checkpoint as in `profile_ou`. Each grid point costs one full empirical-Bayes fit.

```
amp = bliss.profile_ou_amplitude(cum, bb.slope, bb.level, SIG_R=0.5782,
                                phi=0.45, checkpoint="ou_amplitude.pkl")
print("interior maximum at sqrt(V) = %.3f" % amp["argmax"])
for s, d in zip(amp["sqv_grid"], amp["delta"]):
    print("sqrt(V) = %.4f    Delta log-ML = %+.3f" % (s, d))
```

```
interior maximum at sqrt(V) = 0.030
sqrt(V) = 0.0075    Delta log-ML = -2.642
sqrt(V) = 0.0106    Delta log-ML = -2.314
sqrt(V) = 0.0150    Delta log-ML = -1.712
sqrt(V) = 0.0212    Delta log-ML = -0.774
sqrt(V) = 0.0300    Delta log-ML = +0.000
sqrt(V) = 0.0424    Delta log-ML = -1.048
sqrt(V) = 0.0600    Delta log-ML = -4.268
```

The profile carries an interior maximum at $\sqrt{V} = 0.030$: the fixed-shape surface rises monotonically in the amplitude, and the per-point re-equilibration of the noise shape supplies the trade-off that turns the profile over, the converged base scale falling from 0.0281 to 0.0194 across the grid. The standing amplitude 0.0212 sits 0.774 natural-log units below the maximum—twice the margin, 1.55, is below the Kass–Raftery positive grade—so the marginal likelihood prefers 0.030 without distinguishing it from the standing value at that floor, and the fit’s constants stand as given.

7 Reproducibility

Every fit carries a provenance log, a `Provenance` object that records the core-defaults fingerprint—a hash of the certified core-defaults registry, read from the live code objects—so a recorded result is matched to the defaults that produced it. The log also records the forecast window, the calendar specification, and any numerical pivot taken during the optimization. The fit’s results are written to an Excel workbook with `Fit.workbook`, whose Log tab records the author and the creation date, and the figures are written with `Fit.figures`. The analysis of this vignette is reproduced through the package itself on the data shipped under `bliss/data`: the listings of Section 3 run as shown, and `bliss.examples.replicate_paper` regenerates the headline numbers. The replication archive of the accompanying paper is maintained separately and is frozen to the paper.

Bibliography

- Kass, Robert E., and Adrian E. Raftery. "Bayes Factors." *Journal of the American Statistical Association* 90, no. 430 (1995): 773–795.
- Laws, Christopher W., and Frank A. Schmid. "Robust Loss Development Using MCMC: A Vignette." 2010.
- Meyers, Glenn G., and Peng Shi. *The Retrospective Testing of Stochastic Loss Reserve Models*. Casualty Actuarial Society E-Forum, Summer 2011.
- Schmid, Frank A. "Robust Loss Development Using MCMC." *Variance* 4, no. 2 (2010): 100–115.
- Schmid, Frank. "Identification and Bayesian Inference of Loss-Development Structure." Working paper, 2026.