The regression module

LOG EXPECTED COUNT

$$\log(\mu_i) = \alpha + x_i^T \beta$$

COMPLETE EXAMPLE

Pulse counts at two gain settings

```
%%%
Pulse counts in one-second exposures.
%%%
flatppl_compat = "0.1"
reg = standard_module("regression", "0.1")

runs = table(
    level = [-1, 0, 1, -1, 0, 1],
    high = [0, 0, 0, 1, 1, 1],
    counts = [3, 5, 7, 5, 13, 22])
X = colstack([runs.level, runs.high])

alpha ~ Normal(0.0, 2.0)
beta ~ iid(Normal(0.0, 1.0), 2)
y ~ reg.PoissonLogGLM(X, alpha, beta)

prior = lawof(record(alpha = alpha, beta = beta))
K = kernelof(record(y = y), alpha, beta)
L = likelihoodof(K, record(y = runs.counts))
posterior = bayesupdate(L, prior)

eta = alpha .+ X * beta
mean_count = exp.(eta)
predict = functionof(mean_count, alpha, beta)
base_rate = 4.0
rate_ratio = [2.0, 3.0]
point = record(alpha = log(base_rate),
    beta = log.(rate_ratio))
fitted = predict(point)
log_L = logdensityof(L, point)
```

Unordered classes

```
A = [0.0, 0.4, -0.2]
B = rowstack([[0.0, 0.5, 1.0],
              [0.0, 1.0, -0.5]])
label ~ reg.CategoricalLogitGLM(X, A, B)
```

Classes 1–3 each have an intercept in `A` and a column in `B`. Class 1 has zero intercept and slopes, fixing the reference. Softmax converts the scores to probabilities.

Ordered classes

```
grade ~ reg.OrderedLogitGLM(X, beta, [-1, 1])
```

Two increasing cutpoints give three grades. They set the location. For cutpoint t_j, $P(Y_i \le j) = \mathrm{invlogit}(t_j - x_i^T \beta)$.

ANNOTATED LINES

A Rows are runs, columns are predictors 7–11

```
runs = table(
    level = [-1, 0, 1, -1, 0, 1],
    high = [0, 0, 0, 1, 1, 1],
    counts = [3, 5, 7, 5, 13, 22])
X = colstack([runs.level, runs.high])
```

Each row is a one-second run. `colstack` forms a 6×2 matrix from signal level and gain (0 low, 1 high). Pass the intercept separately.

B One law for six counts 13–15

```
alpha ~ Normal(0.0, 2.0)
beta ~ iid(Normal(0.0, 1.0), 2)
y ~ reg.PoissonLogGLM(X, alpha, beta)
```

`beta[1]` multiplies level and `beta[2]` multiplies gain. One undotted call gives a law for six conditionally independent counts with means $\exp(\eta_i)$.

C Condition on the observed counts 17–20

```
prior = lawof(record(alpha = alpha, beta = beta))
K = kernelof(record(y = y), alpha, beta)
L = likelihoodof(K, record(y = runs.counts))
posterior = bayesupdate(L, prior)
```

Make `alpha` and `beta` kernel inputs, then bind the counts to `y` in row order. The prior uses the same parameter fields as the likelihood.

D Evaluate the rate ratios 26–30

```
rate_ratio = [2.0, 3.0]
point = record(alpha = log(base_rate),
    beta = log.(rate_ratio))
fitted = predict(point)
log_L = logdensityof(L, point)
```

A unit increase in level doubles the rate. High gain triples it at any level. Row 5 records 13 counts with mean 12. Its log probability is -2.24838. Summing over all six rows gives `log_L` ≈ -12.16848.

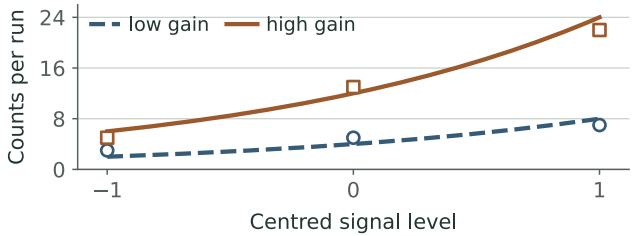
One trial or a fixed trial total

```
hit ~ reg.BernoulliLogitGLM(X, alpha, beta)
hits ~ reg.BinomialLogitGLM(X, 12, alpha, beta)
```

Both use `p = invlogit.(eta)`. Bernoulli gives one binary outcome per row. Binomial counts successes in 12 trials. For a vector of trial totals, use `Binomial.(totals, p)`.

RESPONSE FAMILIES

Expected counts by gain



Curves give the expected counts at `point`. Symbols show the observed counts. At level zero, switching to high gain raises the mean from 4 to 12.

Score a new run

```
X_new = rowstack([[0.5, 1.0]])
new_law = reg.PoissonLogGLM(
    X_new, point.alpha, point.beta)
lp17 = logdensityof(new_law, [17])
```

One run still needs a matrix with one row. At level 0.5 and high gain its mean is $12\sqrt{2} \approx 16.97$. Supply the observed count in a length-one vector.

Continuous readings

```
y_n ~ reg.NormalGLM(X, alpha, beta, 0.5)
y_t ~ reg.StudentTGLM(X, 4, alpha, beta, 0.5)
```

Both laws have location $\eta_i = \alpha + x_i^T \beta$. The Normal standard deviation is 0.5. The Student law has four degrees of freedom and scale 0.5.

Overdispersed counts

```
y_nb ~ reg.NegativeBinomial2LogGLM(
    X, alpha, beta, 4.0)
```

The mean is still $\mu_i = \exp(\eta_i)$. With $\psi = 4$, the variance is $\mu_i + \frac{\mu_i^2}{4}$. A run with mean 12 has variance 48 under this law and 12 under the Poisson law.

Positive readings and fractions

```
delay ~ reg.GammaLogGLM(X, alpha, beta, 3.0)
frac_y ~ reg.BetaLogitGLM(X, alpha, beta, 8.0)
```

The Gamma law has mean $\exp(\eta_i)$ and shape 3. The Beta law has mean $p_i = \mathrm{invlogit}(\eta_i)$ and shape parameters $8p_i$ and $8(1 - p_i)$.

Jessica Cox · bcox@mpp.mpg.de