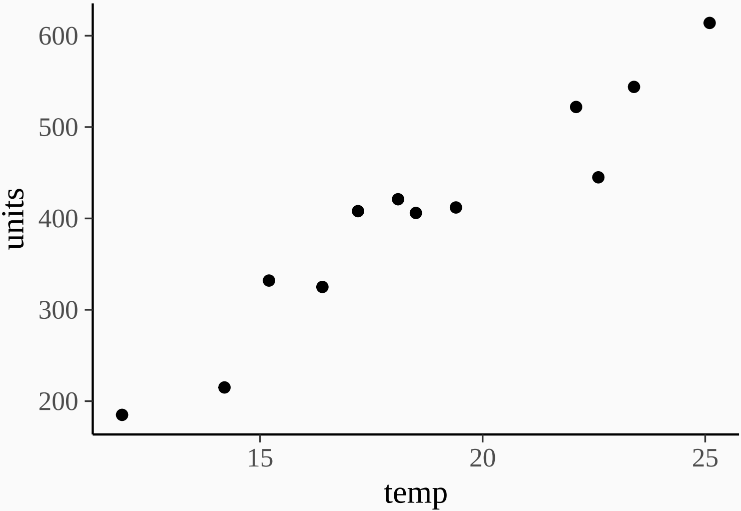
From Classical GLMs to Bayesian MLMs

Paul Bürkner

Part 1: Linear Models

Example: Icecream Sold at Different Temperatures



Thanks to Markus Gesmann!

Linear Models

$$y_n \sim \mathcal{N}(\eta_n, \sigma)$$

$$\eta_n = b_0 + \sum_{i=1}^{k} b_i x_{in}$$

We can write the first expression equivalently as

$$y_n = \eta_n + e_n$$

$$e_n \sim \mathcal{N}(0, \sigma)$$

Writing $y_n \sim \mathcal{N}(\eta_n, \sigma)$ generalizes to other distributions

Fitting Linear Models in R

stats:

```
lm(units ~ temp, data = icecream)
```

rstanarm:

```
stan_lm(units ~ temp, data = icecream)
```

brms:

```
brm(units ~ temp, data = icecream)
```

Differences of rstanarm and brms

rstanarm:

- Emulates existing functions such as `lm`, `glm`, or `glmer`
- Same interface as frequentist counterparts + priors
- Comes with precompiled Stan code
- Saves compilation time when fitting models

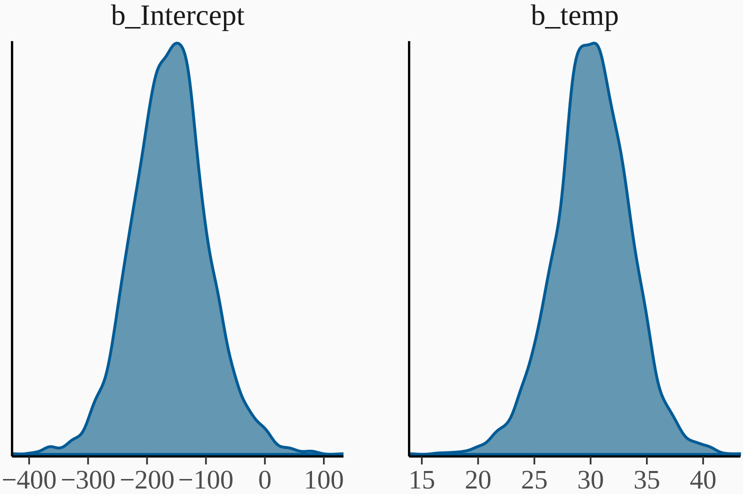
brms:

- One consistent framework for all supported models
- Stan code is written on the fly
- Compilation necessary for every model
- Much greater modeling flexibility
- Somewhat faster for multilevel models

Both are officially supported by the Stan team!

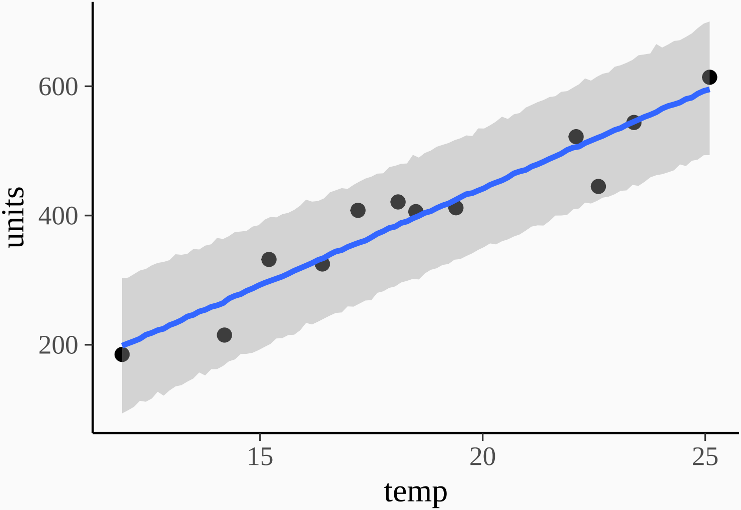
Linear Model: Posterior Distributions

```
stanplot(fit_lin, pars = "^b", type = "dens")
```



Linear Model: Predictions

```
marginal_effects(fit_lin, method = "predict")
```



Assumptions of Linear Models

Validity of the data

- There is no substitute for good data

Statistical assumptions:

- Additivity and linearity
- Independence of errors
- Equal variance of errors
- Normality of errors

Assumptions: Additivity and Linearity

Assume that the predictor term η is a linear combination of the predictor variables multiplied by the regression coefficents:

$$\eta_n = \sum_{i=1}^{K} b_i x_{ni}$$

How to handle violations:

- **User specified non-linear predictor terms**
- Semiparametric methods such as splines or Gaussian processes

Assumptions: Independence of errors

Assume that errors of observations $n \neq m$ are independent

$$p(e_n, e_m) = p(e_n)p(e_m)$$

Equivalently: All signal in the data is captured by the model

Equivalently (for normally distributed errors): Errors are uncorrelated

$$\mathrm{cor}(e_n, e_m) = 0$$

How to handle violations:

- **Find variables explaining the dependency**
- Model errors as correlated

Assumptions: Equal variance and normality of errors

Assume that all errors are normally distributed and share the same variance (or standard deviation):

$$e_n \sim \mathcal{N}(0, \sigma)$$

How to handle violations (equal variances):

- Find variables explaining the unequal variances
- Model unequal variances

How to handle violations (normality of errors):

- Use **transformations** or **generalized linear models**

Modeling Responses on the log-Scale

$$\log(y_n) \sim \mathcal{N}(\eta_n, \sigma)$$

Formula syntax:

```r
log(units) ~ temp
```

Equivalent to a Log-Normal distribution on y:

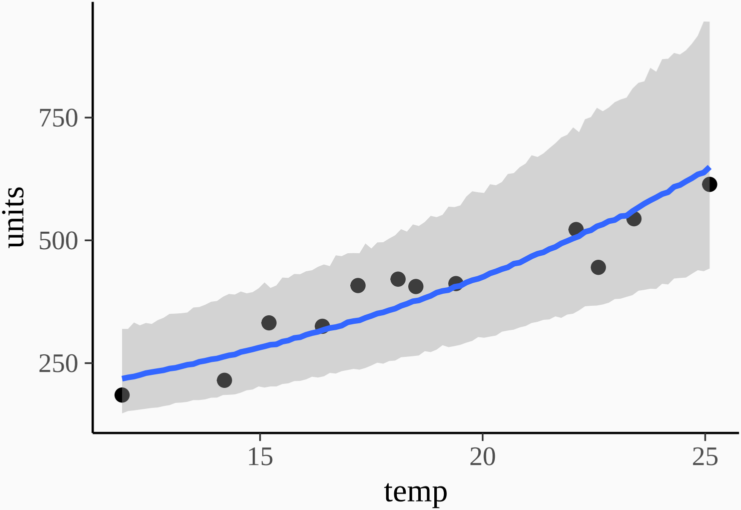
$$y_n \sim \mathcal{LN}(\eta_n, \sigma)$$

Log-Normal model in brms:

```r
brm(units ~ temp, data = icecream, family = lognormal())
```

Lognormal model: Predictions

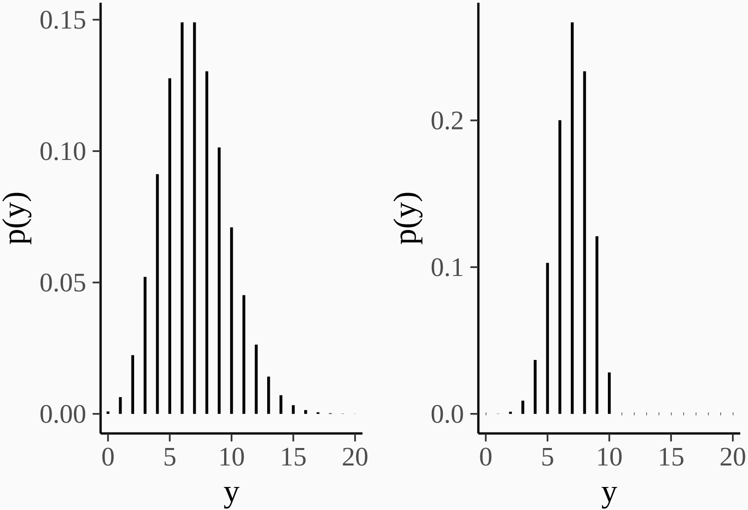
```
marginal_effects(fit_loglin, method = "predict")
```



Part 2: Generalized Linear Models

Generalized Linear Models (GLMs)

Poisson($\lambda = 7$) vs. Binomial($N = 10, \theta = .7$)



Link Functions

The linear predictor is the sum of all effects that are modeled

- Often denoted as η
- In simple linear regression:

$$\eta = b_0 + b_1 x$$

- Directly resembles the mean parameter μ in linear models

Problem:

- The main parameters of non-normal distributions often have a restricted range of definiton such as $\theta \in [0, 1]$ or $\theta \in [0, \infty)$

Solution:

- Define a link function h such that $h(\theta) = \eta$ or equivalently $\theta = g(\eta)$ with response function $g = h^{-1}$

Poisson Models

$$y \sim \text{Poisson}(\lambda) = \frac{\lambda^y \exp(-\lambda)}{y!}$$

$$y \in \{0, 1, ...\}$$

The expected value of the response is modeled as

$$E(y) = \lambda = g(\eta) = \exp(\eta)$$

We call $h = g^{-1}$ the **log** link

Fitting Poisson GLMs in R

stats:

```r
glm(units ~ temp, data = icecream,
    family = poisson("log"))
```

rstanarm:

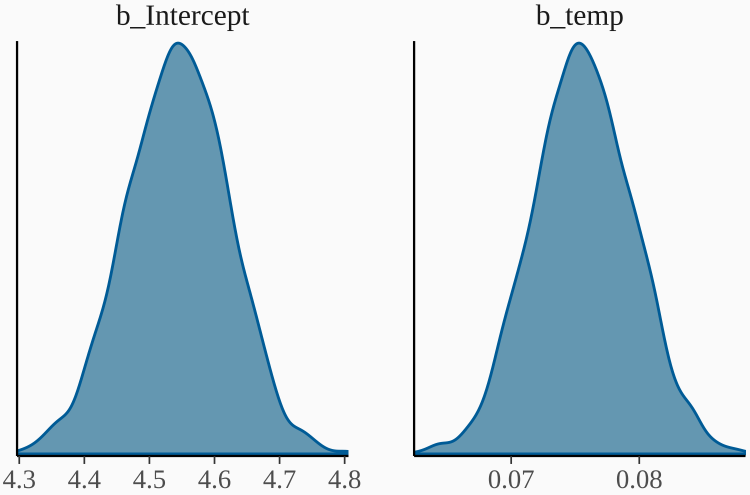
```r
stan_glm(units ~ temp, data = icecream,
         family = poisson("log"))
```

brms:

```r
brm(units ~ temp, data = icecream,
    family = poisson("log"))
```

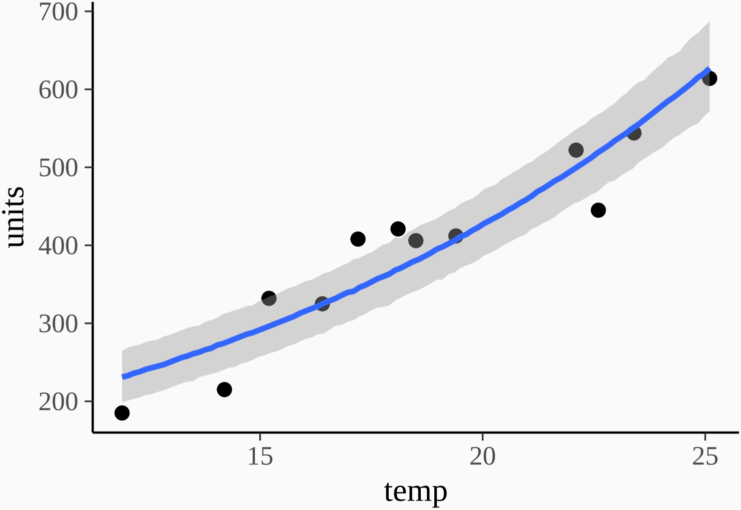
Poisson GLM: Posterior Distribution

```
stanplot(fit_pois, pars = "^b", type = "dens")
```



Poisson GLM: Predictions

```
marginal_effects(fit_pois, method = "predict")
```



Binomial Models

$$y \sim \text{Binomial}(N, \theta) = \binom{N}{y} \theta^y (1 - \theta)^{N-y}$$

$$y \in \{0, 1, ..., N\}$$

The success probability is modeled as

$$\theta = g(\eta) = \frac{\exp(\eta)}{1 + \exp(\eta)}$$

We call $h = g^{-1}$ the **logit** link

The expected value is $E(y) = N\theta = Ng(\eta)$

Fitting Binomial GLMs in R

```
icecream$market_size <- 800
icecream$opportunity <- with(icecream, market_size - units)
```

stats:

```
glm(cbind(units, opportunity) ~ temp, data = icecream,
    family = binomial("logit"))
```

rstanarm:

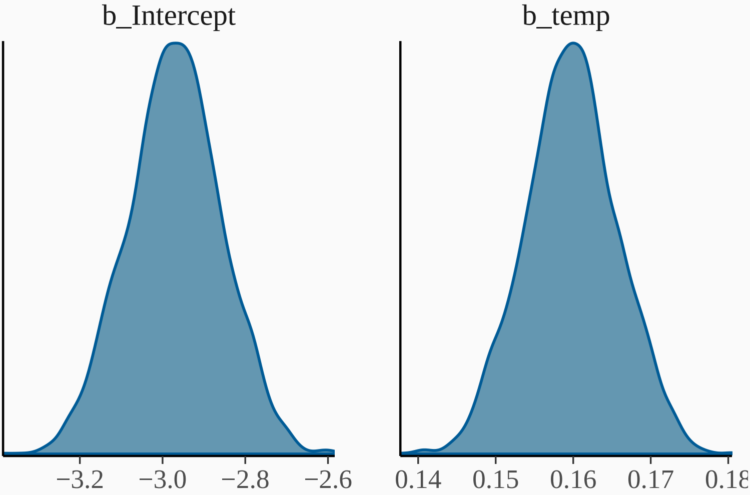
```
stan_glm(cbind(units, opportunity) ~ temp, data = icecream,
         family = binomial("logit"))
```

brms:

```
brm(units | trials(market_size) ~ temp, data = icecream,
    family = binomial("logit"))
```

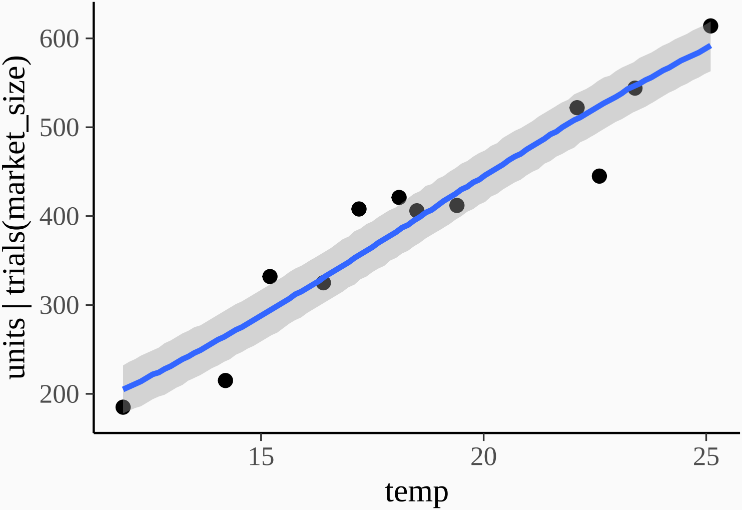
Binomial GLM: Posterior Distribution

```
stanplot(fit_bin, pars = "^b", type = "dens")
```



Binomial GLM: Predictions

```
marginal_effects(fit_bin, method = "predict")
```



Predicted icecream units sold at 35 degrees

```
newdf <- data.frame(temp = 35, market_size = 800)
```

```
predict(fit_lin, newdata = newdf)
predict(fit_loglin, newdata = newdf)
predict(fit_pois, newdata = newdf)
predict(fit_bin, newdata = newdf)
```

model	Estimate	Est.Error	Q2.5	Q97.5
fit_lin	894.3853	72.653244	751.9531	1037.356
fit_loglin	1519.5676	470.263824	859.6983	2541.673
fit_pois	1328.4037	89.701571	1159.0000	1514.000
fit_bin	745.4395	8.767928	728.0000	762.000

Deciding how much icecream to buy

Utility function:

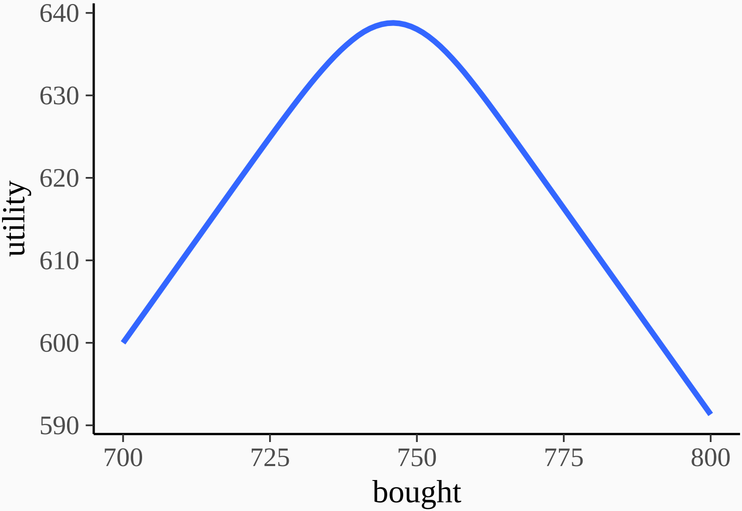
```r
U <- function(units, bought, temp) {
  - 100 - 1 * bought + 2 * pmin(units, bought)
}
```

Utility at 35 degrees when buying 780 units of icecream:

```r
newdf <- data.frame(temp = 35, market_size = 800)
pred <- predict(fit_bin, newdata = newdf, summary = FALSE)
utility <- U(pred, bought = 780, temp = 35)
mean(utility)
```

```
## [1] 611.315
```

Deciding how much icecream to buy: Visualization

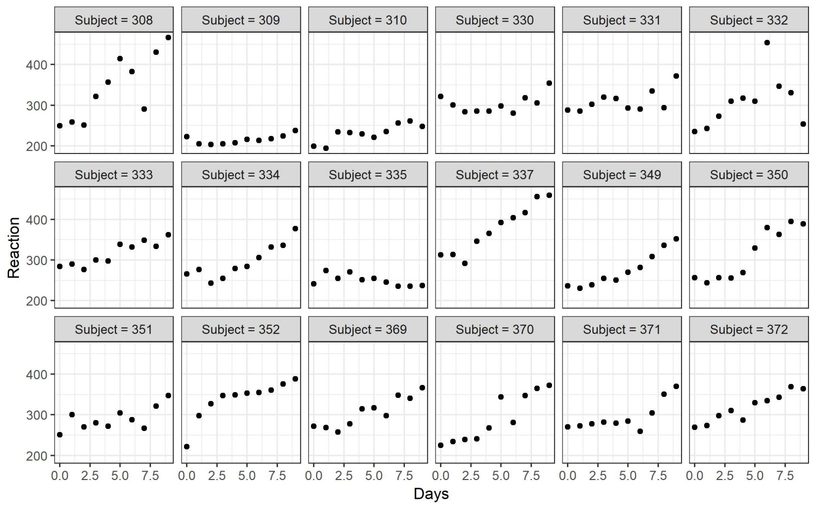


Maximal utility of U = 638.8 at 746 units bought

Part 3: Linear Multilevel Models

Example: Increase in reaction times due to sleep deprivation

```r
data("sleepstudy", package = "lme4")
```



Multilevel Models (MLMs)

- Modeling data measured on different levels within one model
- Account for the dependency structure of the data
- Estimate variation on all levels of the model

Synonyms:

- Hierachical models
- Random effects models
- Mixed (effects) models

Linear MLMs: Varying Intercepts

$$y_n \sim \mathcal{N}(\mu_n, \sigma)$$

$$\mu_n = b_{0j[n]} + b_1 x_n$$

$$b_{0j} \sim \mathcal{N}(b_0, \sigma_{b_0})$$

- Apply shrinkage to the varying (*random*) intercepts
- Assume the slopes to be the same (*fixed*) across participants
- The hierachical prior *partially pools* intercepts

Linear MLMs: Varying Intercepts and Varying Slopes

$$y_n \sim \mathcal{N}(\mu_n, \sigma)$$

$$\mu_n = b_{0j[n]} + b_{1j[n]} x_n$$

$$(b_{0j}, b_{1j}) \sim \mathcal{MN}((b_0, b_1), \Sigma_b)$$

$$\Sigma_b = \begin{pmatrix} \sigma_{b_0}^2 & \sigma_{b_0}\sigma_{b_1}\rho_{b_0 b_1} \\ \sigma_{b_0}\sigma_{b_1}\rho_{b_0 b_1} & \sigma_{b_1}^2 \end{pmatrix}$$

- Apply shrinkage to the varying intercepts and slopes
- Model the correlation between intercepts and slopes
- The hierachical prior *partially pools* intercepts and slopes

Fitting linear MLMs in R

lme4:

```
lmer(Reaction ~ Days + (Days | Subject),
     data = sleepstudy)
```

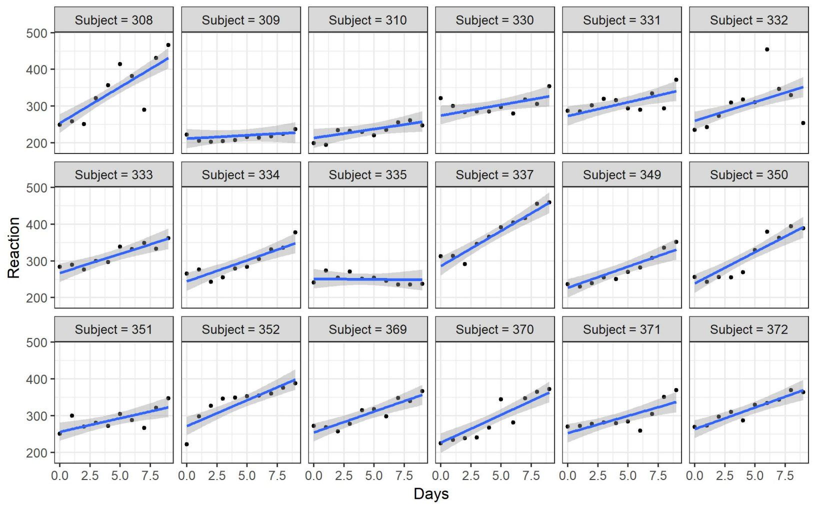
rstanarm:

```
stan_glmer(Reaction ~ Days + (Days | Subject),
           data = sleepstudy)
```

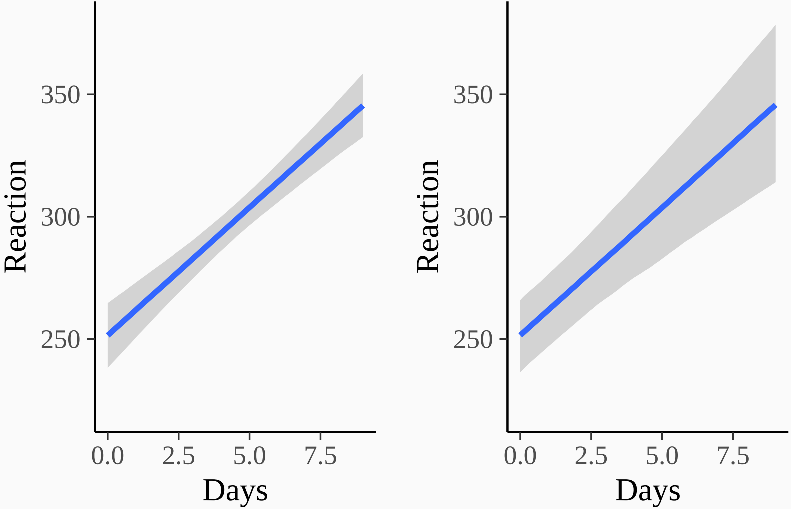
brms:

```
brm(Reaction ~ Days + (Days | Subject),
    data = sleepstudy)
```

Individual Estimates Based on MLMs



LMs vs. MLMs (Complete vs. Partial Pooling)



Some Advantages of Multilevel Models

General:

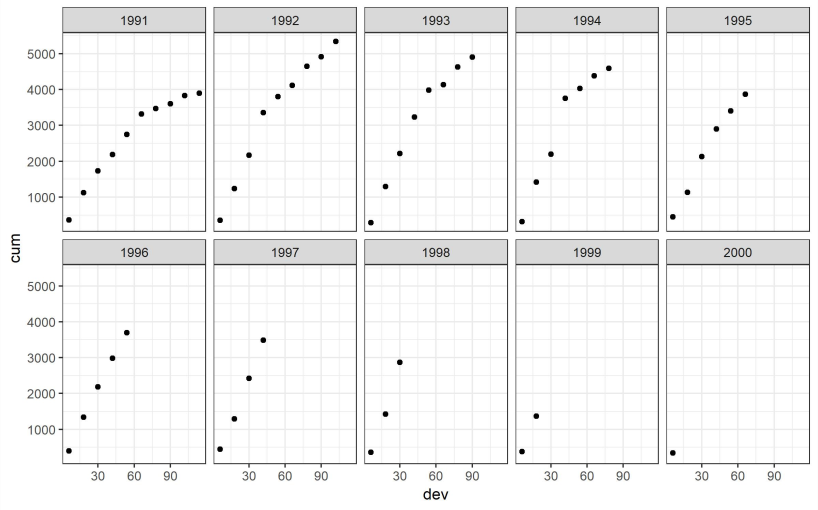
- Conveniently estimate variation on different levels of the data
- Account for all sources of uncertainty
- Increase precision of group-level estimates
- Predict values of new groups not originally present in the data

Bayesian Specifics:

- Greater modeling flexibility
- Improve partial pooling by defining priors on hyperparameters
- Allow to fit more varying effects
- Really estimate varying effects
- Get full posteriors of hierachical parameters

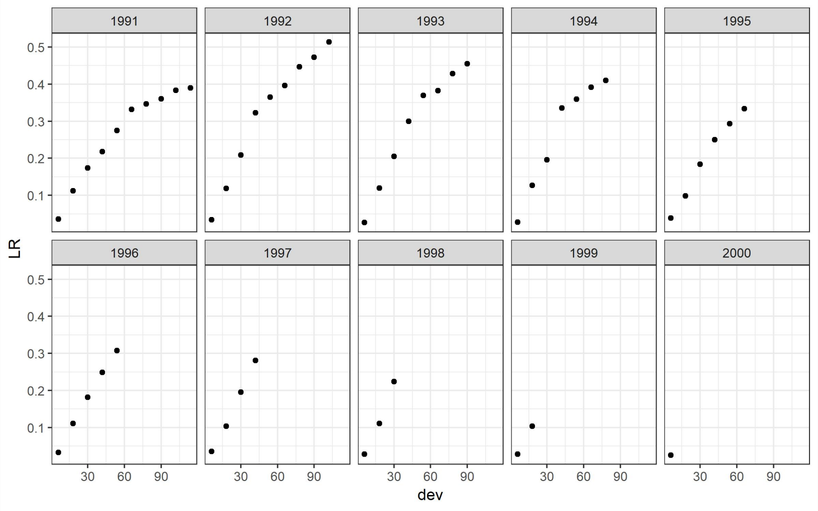
Part 4: Non-Linear Multilevel Models

Example: Cumulative Insurance Loss Payments (Absolute)



Thanks to Markus Gesmann!

Example: Cumulative Insurance Loss Payments (Relative)



Specifying a Non-Linear Model

Growth in loss payments modeled with a weibull curve:

$$G(\,\mathsf{dev}\,|\,\omega,\theta) = 1 - \exp\left(-\left(\frac{\mathsf{dev}}{\theta}\right)^{\omega}\right)$$

Multiply by the ultimate loss ratio (ulr) per accident years (AY):

$$\eta = \mathsf{ulr}_{\mathsf{AY}} \times G(\,\mathsf{dev}\,|\,\omega,\theta)$$

Use a normal distribution for the observed cumulative loss ratios:

$$\mathsf{LR} \sim \mathcal{N}(\eta,\sigma)$$

Other distributions may be reasonable as well

Priors for the Non-Linear Model:

Specify a hierachichal prior on $\mathsf{ulr}_{\mathsf{AY}}$:

$$\mathsf{ulr}_{\mathsf{AY}} \sim \mathcal{N}(\mathsf{ulr}, \sigma_{\mathsf{ulr}})$$

Specify sensible priors on the non-linear parameters:

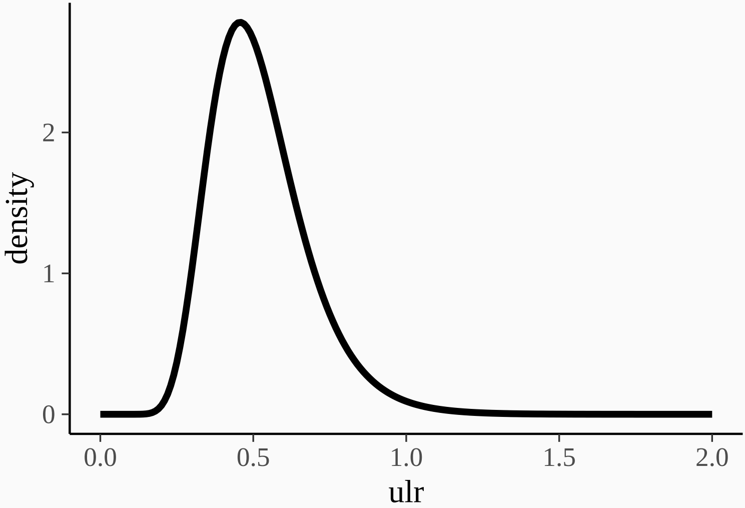
$$\mathsf{ulr} \sim \mathcal{LN}(\log(0.5), 0.3)$$

$$\omega \sim \mathcal{N}_{+}(1, 2)$$

$$\theta \sim \mathcal{N}_{+}(45, 10)$$

Visualize the Prior on the Ultimate Loss

$$\text{ulr} \sim \mathcal{LN}(\log(0.5), 0.3)$$



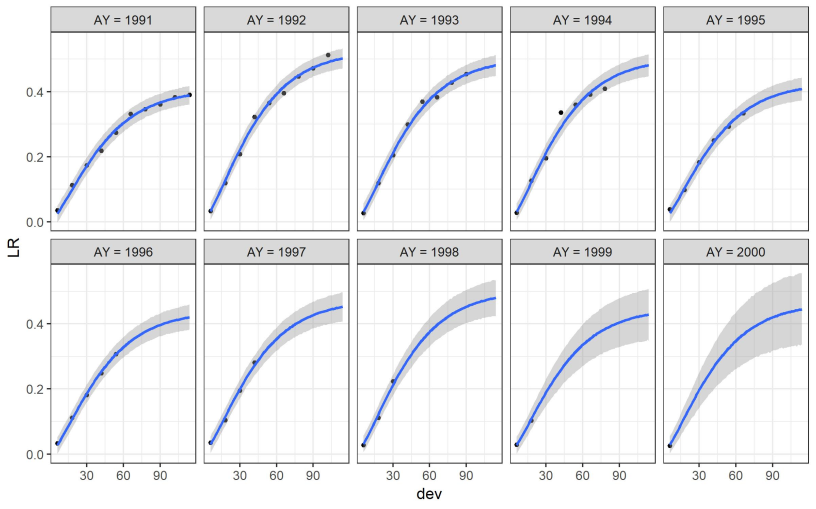
Fitting Non-Linear models with brms

```r
bform <- bf(
  LR ~ ulr * (1 - exp(-(dev/theta)^omega)),
  ulr ~ 1 + (1|AY), omega ~ 1, theta ~ 1,
  nl = TRUE
)

bprior <-
  prior(lognormal(log(0.5), 0.3), nlpar = "ulr", lb = 0) +
  prior(normal(1, 2), nlpar = "omega", lb = 0) +
  prior(normal(45, 10), nlpar = "theta", lb = 0)

fit_LR <- brm(bform, data = loss, prior = bprior)
```

Predictions of Cumulative Loss Payments



Estimation of Ultimate Loss per Accident Year

```
coef(fit_LR)$AY[, , "ulr_Intercept"]
```

	Estimate	Est.Error	Q2.5	Q97.5
1991	0.40	0.01	0.38	0.43
1992	0.52	0.01	0.50	0.55
1993	0.50	0.02	0.47	0.53
1994	0.50	0.02	0.47	0.54
1995	0.43	0.02	0.39	0.46
1996	0.44	0.02	0.40	0.48
1997	0.47	0.02	0.43	0.52
1998	0.50	0.03	0.45	0.56
1999	0.45	0.04	0.37	0.52
2000	0.46	0.06	0.35	0.57

Some Helpful brms Functions (1)

Specify the model using R formulas:

```
brmsformula(formula, ...)
```

Generate the Stan code:

```
make_stancode(formula, ...)
stancode(fit)
```

Generate the data passed to Stan:

```
make_standata(formula, ...)
standata(fit)
```

Handle priors:

```
get_prior(formula, ...)
set_prior(prior, ...)
```

Some Helpful brms Functions (2)

Generate expected values and predictions:

```
fitted(fit, ...)
predict(fit, ...)
marginal_effects(fit, ...)
```

Model comparison:

```
loo(fit1, fit2, ...)
waic(fit1, fit2, ...)
bayes_factor(fit1, fit2, ...)
model_weights(fit1, fit2, ...)
```

Hypothesis testing:

```
hypothesis(fit, hypothesis, ...)
```

Learn more about brms

- Help within R: `help("brms")`
- Vignettes: `vignette(package = "brms")`
- List of all methods: `methods(class = "brmsfit")`
- Website: https://github.com/paul-buerkner/brms
- Forums: http://discourse.mc-stan.org/
- Contact me: `paul.buerkner@gmail.com`
- Twitter: @paulbuerkner

Publications

- Bürkner P. C. (2017). brms: An R Package for Bayesian Multilevel Models using Stan. *Journal of Statistical Software*. 80(1), 1-28. doi:10.18637/jss.v080.i01
- Bürkner P. C. (in press). Advanced Bayesian Multilevel Modeling with the R Package brms. *The R Journal*.

Further Reading

Regression Models:

- Gelman, A., & Hill, J. (2007). *Data analysis using regression and multilevel/hierarchical models.* Cambridge University Press.
- McElreath, R. (2016). *Statistical rethinking: A Bayesian course with examples in R and Stan.* CRC Press.

Stan:

- Carpenter B., Gelman A., Hoffman M. D., Lee D., Goodrich B., Betancourt M., Brubaker M., Guo J., Li P., and Riddell A. (2017). Stan: A probabilistic programming language. *Journal of Statistical Software*. 76(1). 10.18637/jss.v076.i01
- User Manual: http://mc-stan.org/users/documentation