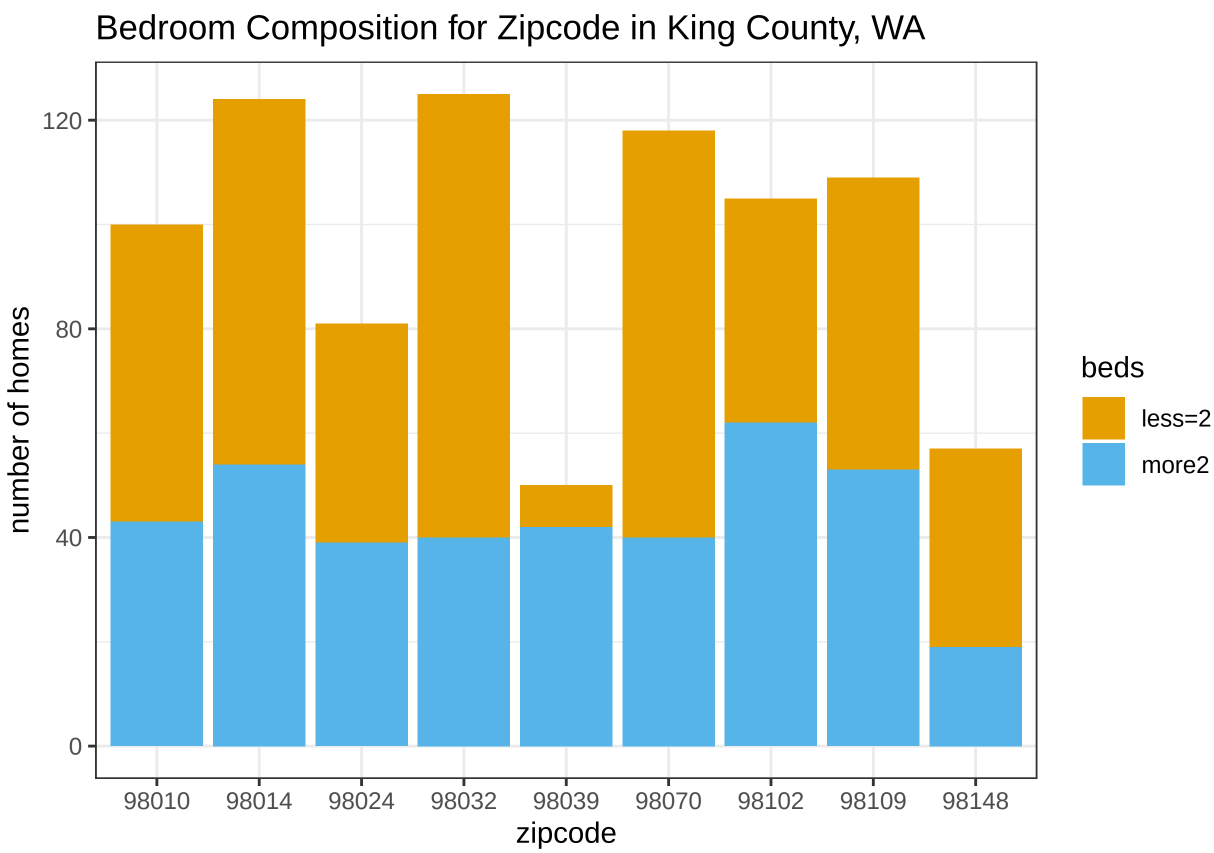
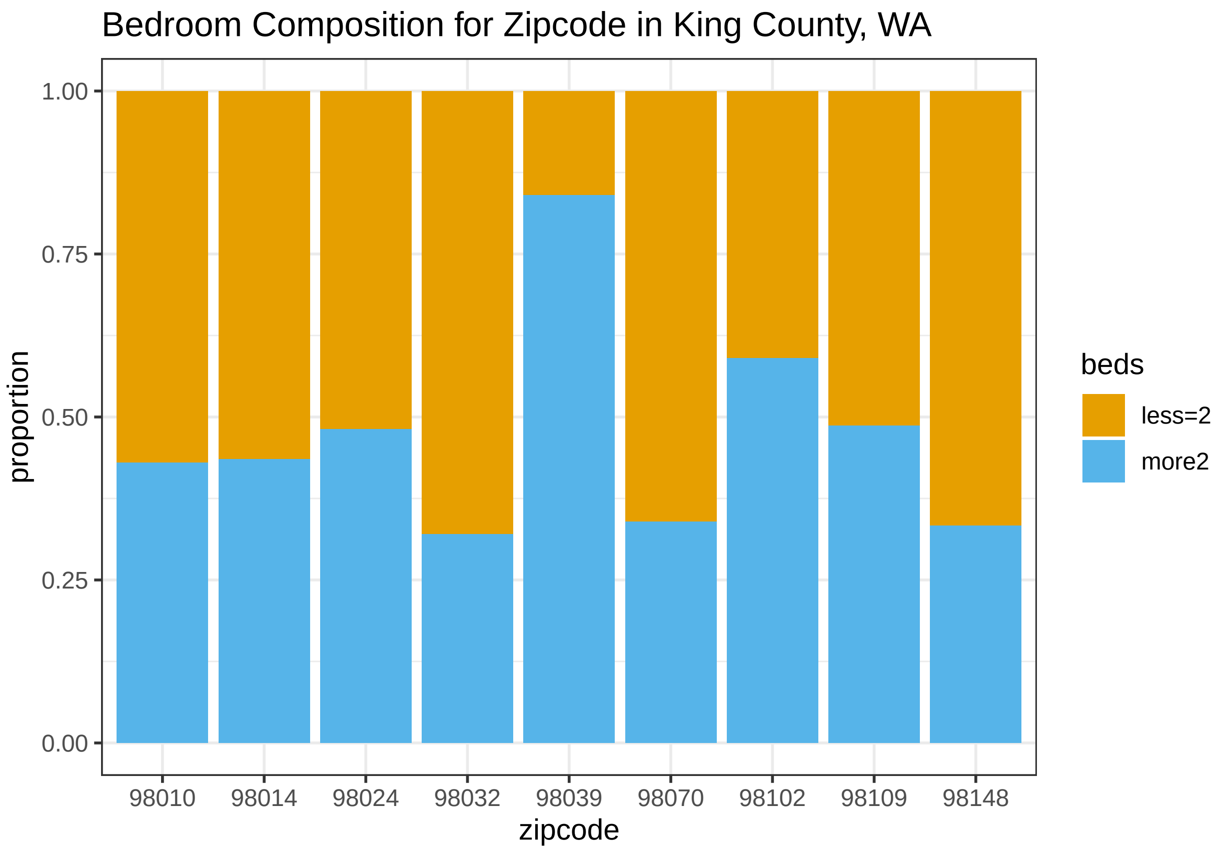
Hierarchical GLMs





Hierarchical GLMs

Multilevel principles (different effects across different groups) can also be applied to GLMs.

Consider a multilevel - logistic regression model:

$$
\begin{aligned}
y_i &\sim Bernoulli(p_i) \\
logit(p_i) &= \alpha_{j[i]} + \beta_{j[i]} \\
\alpha_j &\sim N(\mu_\alpha, \sigma_\alpha^2) \\
\beta_j &\sim N(\mu_\beta, \sigma_\beta^2)
\end{aligned}
$$

```r
glm1 <- glm(cbind(more2,lessequal2) ~ 1, data = seattle, family = binomial)
display(glm1)
```

```
## glm(formula = cbind(more2, lessequal2) ~ 1, family = binomial,
##     data = seattle)
##             coef.est coef.se
## (Intercept) -0.20     0.07
## ---
##   n = 869, k = 1
##   residual deviance = 1196.4, null deviance = 1196.4 (difference = 0.0)
```

```r
invlogit(coef(glm1))
```

```
## (Intercept)
##   0.4510932
```

```r
stan1 <- stan_glm(cbind(more2,lessequal2) ~ 1, data = seattle, family = binomial, refresh = 0)
print(stan1)
```

```
## stan_glm
##  family:       binomial [logit]
##  formula:      cbind(more2, lessequal2) ~ 1
##  observations: 869
##  predictors:   1
## ------
##             Median MAD_SD
## (Intercept) -0.2    0.1
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

```r
invlogit(coef(stan1))
```

```
## (Intercept)
##   0.4512607
```

```r
seattle %>% summarise(freq = mean(more2))
```

```
## # A tibble: 1 x 1
##    freq
##   <dbl>
## 1 0.451
```

```r
glmer1 <- stan_glmer(cbind(more2,lessequal2) ~ 1 + (1 | zipcode),
                     data = seattle, family = binomial, refresh = 0)
print(glmer1)
```

```
## stan_glmer
##  family:       binomial [logit]
##  formula:      cbind(more2, lessequal2) ~ 1 + (1 | zipcode)
##  observations: 869
## ------
##             Median MAD_SD
## (Intercept) -0.1    0.2  
## 
## Error terms:
##  Groups  Name        Std.Dev.
##  zipcode (Intercept) 0.76    
## Num. levels: zipcode 9 
## 
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

```r
fixef(glmer1)
```

```
## (Intercept) 
##  -0.1109277
```

```r
ranef(glmer1)
```

```
## $zipcode
##       (Intercept)
## 98010 -0.15399026
## 98014 -0.14166485
## 98024  0.02841332
## 98032 -0.59365888
## 98039  1.36326653
## 98070 -0.50633806
## 98102  0.43013463
## 98109  0.05408178
## 98148 -0.49481567
## 
## with conditional variances for "zipcode"
```

```r
coef(glmer1)
```

```
## $zipcode
##       (Intercept)
## 98010 -0.26491795
## 98014 -0.25259254
## 98024 -0.08251437
## 98032 -0.70458657
## 98039  1.25233884
## 98070 -0.61726575
## 98102  0.31920694
## 98109 -0.05684591
## 98148 -0.60574336
## 
```

```
## attr(,"class")
## [1] "coef.mer"
```

```
seattle %>% group_by(zipcode) %>% summarise(freq = mean(more2), n = n()) %>%
  ungroup() %>%
  bind_cols(tibble(glmer_est = invlogit(coef(glmer1)$zipcode[[1]]))
)
```

```
## # A tibble: 9 x 4
##   zipcode  freq     n glmer_est
##   <fct>   <dbl> <int>     <dbl>
## 1 98010   0.43    100     0.434
## 2 98014   0.435   124     0.437
## 3 98024   0.481    81     0.479
## 4 98032   0.32    125     0.331
## 5 98039   0.84     50     0.778
## 6 98070   0.339   118     0.350
## 7 98102   0.590   105     0.579
## 8 98109   0.486   109     0.486
## 9 98148   0.333    57     0.353
```

Covariates can also be added that vary across the groups

```r
glmer2 <- stan_glmer(cbind(more2,lessequal2) ~  scale_sqft + (1 + scale_sqft | zipcode),
                data = seattle, family = binomial, refresh = 0)
print(glmer2)
```

```
## stan_glmer
##  family:       binomial [logit]
##  formula:      cbind(more2, lessequal2) ~ scale_sqft + (1 + scale_sqft | zipcode)
##  observations: 869
## ------
##             Median MAD_SD
## (Intercept) 0.0    0.1
## scale_sqft  2.5    0.3
##
## Error terms:
##  Groups  Name        Std.Dev. Corr
##  zipcode (Intercept) 0.36
##          scale_sqft  0.71     -0.46
## Num. levels: zipcode 9
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

```r
fixef(glmer2)
```

```
## (Intercept)  scale_sqft
## 0.005142038 2.546675394
```

```r
ranef(glmer2)
```

```
## $zipcode
##       (Intercept)  scale_sqft
## 98010 -0.19063716  0.33425669
## 98014 -0.11578616  0.37992182
## 98024 -0.09710945  0.31738982
## 98032 -0.04653161 -0.05084137
## 98039  0.02035965  0.01294492
## 98070 -0.21510327  0.29402616
## 98102  0.44798899 -1.16330372
## 98109  0.12861834 -0.27353437
## 98148  0.06126657  0.24316386
##
## with conditional variances for "zipcode"
```

```r
coef(glmer2)
```

```
## $zipcode
##       (Intercept) scale_sqft
## 98010 -0.18549512   2.880932
## 98014 -0.11064412   2.926597
## 98024 -0.09196741   2.864065
## 98032 -0.04138957   2.495834
## 98039  0.02550169   2.559620
## 98070 -0.20996123   2.840702
## 98102  0.45313103   1.383372
```

```
## 98109  0.13376038   2.273141
## 98148  0.06640861   2.789839
##
## attr(,"class")
## [1] "coef.mer"
```

Stan

Stan is a more general approach for fitting hierarchical models. Both have additional flexibility for specifying sampling models directly.

Below is the syntax for a Stan model for hierarchical logistic regression.

```
data {
  int<lower=1> D;
  int<lower=0> N;
  int<lower=1> L;
  int<lower=0,upper=1> y[N];
  int<lower=1,upper=L> ll[N];
  row_vector[D] x[N];
}
parameters {
  real mu[D];
  real<lower=0> sigma[D];
  vector[D] beta[L];
}
model {
  for (d in 1:D) {
    mu[d] ~ normal(0, 100);
    for (l in 1:L)
      beta[l,d] ~ normal(mu[d], sigma[d]);
  }
  for (n in 1:N)
    y[n] ~ bernoulli(inv_logit(x[n] * beta[ll[n]]));
}
```