Simulate a binomial response and random effects

Eamonn O'Brien

17 July, 2019

Contents

Contents

List of Figures

List of Tables

Some functions

Introduction

```
cat("\nI simulate between country and between site SD with an underlying mean (see reference where code
```

```
I simulate between country and between site SD with an underlying mean (see reference where code was hel
```

Population parameters & Design Paramters for simulation

```
# mu: underlying mean of the outcome in the control group
# beta1: covariate not used sdcountry: sd of random effect at
# the country level (sd for the bi) sdsite: sd of random
# effect at the site level (sd for the bij)

# Design parameters countries: number of countries sites:
# number of sites per country persons : number of persons per
# site

mu = 0.8  # 0.5 means odds = 1, so intercept should be zero (log odds)
(reg.intercept <- log(mu/(1 - mu)))  #   the intercept of the regression model should approximate this
```

```
[1] 1.386294
```

```
# get back to prob
exp(reg.intercept)/(1 + exp(reg.intercept))  # convert log odds to prob
```

```
[1] 0.8
```

```
beta1 = 1  # if this was 1, means no difference, log(1)==0 * trt

sdcountry = 0.2  # this is on the log odds scale
sdsite = 0.4  # this is on the log odds scale

# Design Parameters
countries = 25  # no of countries

# no of sites in countries
sites <- MASS::rnegbin(countries, mu = 10, theta = 1.6)
sites <- ifelse(sites == 0, 1, sites)  # dont want 0s

# no persons at each site
persons <- MASS::rnegbin(sum(sites), mu = 11, theta = 2) + 1  # dont want any 0s hence the + 1
```

Examine simulation data

```
sum(persons)
```

```
[1] 2090
```

```
sites  # shows the number of sites in each of the countries
```

```
 [1]  3 20  1  7 25 11  1  1 15  6  8  9  3 21 12 10  7  3
[19]  1  7  1  2  1  3  7
```

```r
sequence(sites)  # expand the sites
```

```
  [1]  1  2  3  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
 [19] 16 17 18 19 20  1  1  2  3  4  5  6  7  1  2  3  4  5
 [37]  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
 [55] 24 25  1  2  3  4  5  6  7  8  9 10 11  1  1  1  2  3
 [73]  4  5  6  7  8  9 10 11 12 13 14 15  1  2  3  4  5  6
 [91]  1  2  3  4  5  6  7  8  1  2  3  4  5  6  7  8  9  1
[109]  2  3  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16
[127] 17 18 19 20 21  1  2  3  4  5  6  7  8  9 10 11 12  1
[145]  2  3  4  5  6  7  8  9 10  1  2  3  4  5  6  7  1  2
[163]  3  1  1  2  3  4  5  6  7  1  1  2  1  1  2  3  1  2
[181]  3  4  5  6  7
```

```r
persons  # shows the persons at each site
```

```
  [1]  3 11 15 14 11  6  1  2 12  9  8 24  2 12  5 12 21 39
 [19]  9  6  7 12  8  2  7  8 10  9  5 13 15 15  8 13  9 10
 [37]  3 21  3 20 11 10 26 13 24  6  8 16  9  4  9  9  3 13
 [55]  9 13  4 16 18 15 17 13 26  7  3  8  4 14 17 14 11  7
 [73] 17  2 15 10 16  7  2 31 11 17  9  7 19  1 14 14  8 14
 [91]  4 44  7 18  3 10  7  4  7 14 14 10 14  4 10 16 24  8
[109] 22 12  4  2 35 22  7  1  2  7 30  9  9  6  3 15 10 35
[127]  9  9  7  5  8 10  9  3 12  3 10  4  6 26 25  2  6 15
[145] 14  4  9  2  1  4  8 30 24 12  2  6 11 14 11 20 41 16
[163]  4 15 18  4 12 10  2 10 12  4 14 10  1  7  9 14 19  6
[181]  6  9 20 19  7
```

```r
rep(1:length(sites), sites)  # grouping indicator for sites
```

```
  [1]  1  1  1  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2
 [19]  2  2  2  2  2  3  4  4  4  4  4  4  4  5  5  5  5  5
 [37]  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5  5
 [55]  5  5  6  6  6  6  6  6  6  6  6  6  6  7  8  9  9  9
 [73]  9  9  9  9  9  9  9  9  9  9  9  9 10 10 10 10 10 10
 [91] 11 11 11 11 11 11 11 11 12 12 12 12 12 12 12 12 12 13
[109] 13 13 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14
[127] 14 14 14 14 14 15 15 15 15 15 15 15 15 15 15 15 15 16
[145] 16 16 16 16 16 16 16 16 16 17 17 17 17 17 17 17 18 18
[163] 18 19 20 20 20 20 20 20 20 21 22 22 23 24 24 24 25 25
[181] 25 25 25 25 25
```

Create unbalanced data. Trial and error to program this!

```r
# pp <- seq_along(rep(persons, persons))# count of persons
# (not really needed)
pp <- rep(1:sum(persons))  # count of persons (not really needed) but simpler
g <- rep(1:length(persons), persons)  # person in each site is 'flattened'
x <- rep(1:length(sites), sites)  # sites ditto
country <- rep(x, persons)  # countries

# put tx in there for now although it does not vary
tx = 1
d <- cbind(country = country, site = g, person = pp, tx = tx)  # create a data frame
summary(d)
```

```
    country          site            person      
 Min.   : 1.0   Min.   :  1.00   Min.   :   1.0  
 1st Qu.: 5.0   1st Qu.: 48.00   1st Qu.: 523.2  
 Median :11.0   Median : 92.00   Median :1045.5  
 Mean   :10.9   Mean   : 93.51   Mean   :1045.5  
 3rd Qu.:15.0   3rd Qu.:140.00   3rd Qu.:1567.8  
 Max.   :25.0   Max.   :185.00   Max.   :2090.0  
       tx   
 Min.   :1  
 1st Qu.:1  
 Median :1  
 Mean   :1  
 3rd Qu.:1  
 Max.   :1  
```

```
# draw random effects for clusters
countryRE <- rnorm(countries, 0, sdcountry)
siteRE <- rnorm(sum(sites), 0, sdsite)


# create outcome
(prob <- 1/(1 + exp(-(log(mu/(1 - mu)) + log(beta1) * tx + countryRE[d[,
    1]] + siteRE[d[, 2]]))))
```

```
   [1] 0.7068401 0.7068401 0.7068401 0.7089336 0.7089336
   [6] 0.7089336 0.7089336 0.7089336 0.7089336 0.7089336
  [11] 0.7089336 0.7089336 0.7089336 0.7089336 0.7381740
  [16] 0.7381740 0.7381740 0.7381740 0.7381740 0.7381740
  [21] 0.7381740 0.7381740 0.7381740 0.7381740 0.7381740
  [26] 0.7381740 0.7381740 0.7381740 0.7381740 0.8511869
  [31] 0.8511869 0.8511869 0.8511869 0.8511869 0.8511869
  [36] 0.8511869 0.8511869 0.8511869 0.8511869 0.8511869
  [41] 0.8511869 0.8511869 0.8511869 0.7167832 0.7167832
  [46] 0.7167832 0.7167832 0.7167832 0.7167832 0.7167832
  [51] 0.7167832 0.7167832 0.7167832 0.7167832 0.7699839
  [56] 0.7699839 0.7699839 0.7699839 0.7699839 0.7699839
  [61] 0.7247097 0.7440638 0.7440638 0.7657813 0.7657813
  [66] 0.7657813 0.7657813 0.7657813 0.7657813 0.7657813
  [71] 0.7657813 0.7657813 0.7657813 0.7657813 0.7657813
  [76] 0.8306589 0.8306589 0.8306589 0.8306589 0.8306589
  [81] 0.8306589 0.8306589 0.8306589 0.8306589 0.7516713
  [86] 0.7516713 0.7516713 0.7516713 0.7516713 0.7516713
  [91] 0.7516713 0.7516713 0.7931725 0.7931725 0.7931725
  [96] 0.7931725 0.7931725 0.7931725 0.7931725 0.7931725
 [101] 0.7931725 0.7931725 0.7931725 0.7931725 0.7931725
 [106] 0.7931725 0.7931725 0.7931725 0.7931725 0.7931725
 [111] 0.7931725 0.7931725 0.7931725 0.7931725 0.7931725
 [116] 0.7931725 0.7974710 0.7974710 0.8061098 0.8061098
 [121] 0.8061098 0.8061098 0.8061098 0.8061098 0.8061098
 [126] 0.8061098 0.8061098 0.8061098 0.8061098 0.8061098
 [131] 0.6961986 0.6961986 0.6961986 0.6961986 0.6961986
 [136] 0.8468066 0.8468066 0.8468066 0.8468066 0.8468066
 [141] 0.8468066 0.8468066 0.8468066 0.8468066 0.8468066
 [146] 0.8468066 0.8468066 0.7123113 0.7123113 0.7123113
 [151] 0.7123113 0.7123113 0.7123113 0.7123113 0.7123113
```

```
[156] 0.7123113 0.7123113 0.7123113 0.7123113 0.7123113
[161] 0.7123113 0.7123113 0.7123113 0.7123113 0.7123113
[166] 0.7123113 0.7123113 0.7123113 0.7343302 0.7343302
[171] 0.7343302 0.7343302 0.7343302 0.7343302 0.7343302
[176] 0.7343302 0.7343302 0.7343302 0.7343302 0.7343302
[181] 0.7343302 0.7343302 0.7343302 0.7343302 0.7343302
[186] 0.7343302 0.7343302 0.7343302 0.7343302 0.7343302
[191] 0.7343302 0.7343302 0.7343302 0.7343302 0.7343302
[196] 0.7343302 0.7343302 0.7343302 0.7343302 0.7343302
[201] 0.7343302 0.7343302 0.7343302 0.7343302 0.7343302
[206] 0.7343302 0.7343302 0.8348832 0.8348832 0.8348832
[211] 0.8348832 0.8348832 0.8348832 0.8348832 0.8348832
[216] 0.8348832 0.8287298 0.8287298 0.8287298 0.8287298
[221] 0.8287298 0.8287298 0.6509055 0.6509055 0.6509055
[226] 0.6509055 0.6509055 0.6509055 0.6509055 0.6703548
[231] 0.6703548 0.6703548 0.6703548 0.6703548 0.6703548
[236] 0.6703548 0.6703548 0.6703548 0.6703548 0.6703548
[241] 0.6703548 0.6261621 0.6261621 0.6261621 0.6261621
[246] 0.6261621 0.6261621 0.6261621 0.6261621 0.8513354
[251] 0.8513354 0.8404468 0.8404468 0.8404468 0.8404468
[256] 0.8404468 0.8404468 0.8404468 0.8508478 0.8508478
[261] 0.8508478 0.8508478 0.8508478 0.8508478 0.8508478
[266] 0.8508478 0.8601436 0.8601436 0.8601436 0.8601436
[271] 0.8601436 0.8601436 0.8601436 0.8601436 0.8601436
[276] 0.8601436 0.7949235 0.7949235 0.7949235 0.7949235
[281] 0.7949235 0.7949235 0.7949235 0.7949235 0.7949235
[286] 0.7681939 0.7681939 0.7681939 0.7681939 0.7681939
[291] 0.8814605 0.8814605 0.8814605 0.8814605 0.8814605
[296] 0.8814605 0.8814605 0.8814605 0.8814605 0.8814605
[301] 0.8814605 0.8814605 0.8814605 0.8694432 0.8694432
[306] 0.8694432 0.8694432 0.8694432 0.8694432 0.8694432
[311] 0.8694432 0.8694432 0.8694432 0.8694432 0.8694432
[316] 0.8694432 0.8694432 0.8694432 0.7046803 0.7046803
[321] 0.7046803 0.7046803 0.7046803 0.7046803 0.7046803
[326] 0.7046803 0.7046803 0.7046803 0.7046803 0.7046803
[331] 0.7046803 0.7046803 0.7046803 0.8077247 0.8077247
[336] 0.8077247 0.8077247 0.8077247 0.8077247 0.8077247
[341] 0.8077247 0.7530551 0.7530551 0.7530551 0.7530551
[346] 0.7530551 0.7530551 0.7530551 0.7530551 0.7530551
[351] 0.7530551 0.7530551 0.7530551 0.7530551 0.6558945
[356] 0.6558945 0.6558945 0.6558945 0.6558945 0.6558945
[361] 0.6558945 0.6558945 0.6558945 0.8225033 0.8225033
[366] 0.8225033 0.8225033 0.8225033 0.8225033 0.8225033
[371] 0.8225033 0.8225033 0.8225033 0.8087127 0.8087127
[376] 0.8087127 0.8075867 0.8075867 0.8075867 0.8075867
[381] 0.8075867 0.8075867 0.8075867 0.8075867 0.8075867
[386] 0.8075867 0.8075867 0.8075867 0.8075867 0.8075867
[391] 0.8075867 0.8075867 0.8075867 0.8075867 0.8075867
[396] 0.8075867 0.8075867 0.8263265 0.8263265 0.8263265
[401] 0.8613249 0.8613249 0.8613249 0.8613249 0.8613249
[406] 0.8613249 0.8613249 0.8613249 0.8613249 0.8613249
[411] 0.8613249 0.8613249 0.8613249 0.8613249 0.8613249
[416] 0.8613249 0.8613249 0.8613249 0.8613249 0.8613249
[421] 0.8257198 0.8257198 0.8257198 0.8257198 0.8257198
```

```
[426] 0.8257198 0.8257198 0.8257198 0.8257198 0.8257198
[431] 0.8257198 0.7732541 0.7732541 0.7732541 0.7732541
[436] 0.7732541 0.7732541 0.7732541 0.7732541 0.7732541
[441] 0.7732541 0.7648457 0.7648457 0.7648457 0.7648457
[446] 0.7648457 0.7648457 0.7648457 0.7648457 0.7648457
[451] 0.7648457 0.7648457 0.7648457 0.7648457 0.7648457
[456] 0.7648457 0.7648457 0.7648457 0.7648457 0.7648457
[461] 0.7648457 0.7648457 0.7648457 0.7648457 0.7648457
[466] 0.7648457 0.7648457 0.8054373 0.8054373 0.8054373
[471] 0.8054373 0.8054373 0.8054373 0.8054373 0.8054373
[476] 0.8054373 0.8054373 0.8054373 0.8054373 0.8054373
[481] 0.8316489 0.8316489 0.8316489 0.8316489 0.8316489
[486] 0.8316489 0.8316489 0.8316489 0.8316489 0.8316489
[491] 0.8316489 0.8316489 0.8316489 0.8316489 0.8316489
[496] 0.8316489 0.8316489 0.8316489 0.8316489 0.8316489
[501] 0.8316489 0.8316489 0.8316489 0.8316489 0.7422087
[506] 0.7422087 0.7422087 0.7422087 0.7422087 0.7422087
[511] 0.8021249 0.8021249 0.8021249 0.8021249 0.8021249
[516] 0.8021249 0.8021249 0.8021249 0.8653271 0.8653271
[521] 0.8653271 0.8653271 0.8653271 0.8653271 0.8653271
[526] 0.8653271 0.8653271 0.8653271 0.8653271 0.8653271
[531] 0.8653271 0.8653271 0.8653271 0.8653271 0.8069076
[536] 0.8069076 0.8069076 0.8069076 0.8069076 0.8069076
[541] 0.8069076 0.8069076 0.8069076 0.7674528 0.7674528
[546] 0.7674528 0.7674528 0.7962426 0.7962426 0.7962426
[551] 0.7962426 0.7962426 0.7962426 0.7962426 0.7962426
[556] 0.7962426 0.8720505 0.8720505 0.8720505 0.8720505
[561] 0.8720505 0.8720505 0.8720505 0.8720505 0.8720505
[566] 0.8446668 0.8446668 0.8446668 0.8773985 0.8773985
[571] 0.8773985 0.8773985 0.8773985 0.8773985 0.8773985
[576] 0.8773985 0.8773985 0.8773985 0.8773985 0.8773985
[581] 0.8773985 0.8218587 0.8218587 0.8218587 0.8218587
[586] 0.8218587 0.8218587 0.8218587 0.8218587 0.8218587
[591] 0.8371893 0.8371893 0.8371893 0.8371893 0.8371893
[596] 0.8371893 0.8371893 0.8371893 0.8371893 0.8371893
[601] 0.8371893 0.8371893 0.8371893 0.7868342 0.7868342
[606] 0.7868342 0.7868342 0.8546414 0.8546414 0.8546414
[611] 0.8546414 0.8546414 0.8546414 0.8546414 0.8546414
[616] 0.8546414 0.8546414 0.8546414 0.8546414 0.8546414
[621] 0.8546414 0.8546414 0.8546414 0.7903103 0.7903103
[626] 0.7903103 0.7903103 0.7903103 0.7903103 0.7903103
[631] 0.7903103 0.7903103 0.7903103 0.7903103 0.7903103
[636] 0.7903103 0.7903103 0.7903103 0.7903103 0.7903103
[641] 0.7903103 0.7233428 0.7233428 0.7233428 0.7233428
[646] 0.7233428 0.7233428 0.7233428 0.7233428 0.7233428
[651] 0.7233428 0.7233428 0.7233428 0.7233428 0.7233428
[656] 0.7233428 0.8292792 0.8292792 0.8292792 0.8292792
[661] 0.8292792 0.8292792 0.8292792 0.8292792 0.8292792
[666] 0.8292792 0.8292792 0.8292792 0.8292792 0.8292792
[671] 0.8292792 0.8292792 0.8292792 0.9095103 0.9095103
[676] 0.9095103 0.9095103 0.9095103 0.9095103 0.9095103
[681] 0.9095103 0.9095103 0.9095103 0.9095103 0.9095103
[686] 0.9095103 0.8640914 0.8640914 0.8640914 0.8640914
[691] 0.8640914 0.8640914 0.8640914 0.8640914 0.8640914
```

```
[696] 0.8640914 0.8640914 0.8640914 0.8640914 0.8640914
[701] 0.8640914 0.8640914 0.8640914 0.8640914 0.8640914
[706] 0.8640914 0.8640914 0.8640914 0.8640914 0.8640914
[711] 0.8640914 0.8640914 0.8803166 0.8803166 0.8803166
[716] 0.8803166 0.8803166 0.8803166 0.8803166 0.8419584
[721] 0.8419584 0.8419584 0.7834536 0.7834536 0.7834536
[726] 0.7834536 0.7834536 0.7834536 0.7834536 0.7834536
[731] 0.8097573 0.8097573 0.8097573 0.8097573 0.8115984
[736] 0.8115984 0.8115984 0.8115984 0.8115984 0.8115984
[741] 0.8115984 0.8115984 0.8115984 0.8115984 0.8115984
[746] 0.8115984 0.8115984 0.8115984 0.6607481 0.6607481
[751] 0.6607481 0.6607481 0.6607481 0.6607481 0.6607481
[756] 0.6607481 0.6607481 0.6607481 0.6607481 0.6607481
[761] 0.6607481 0.6607481 0.6607481 0.6607481 0.6607481
[766] 0.8687574 0.8687574 0.8687574 0.8687574 0.8687574
[771] 0.8687574 0.8687574 0.8687574 0.8687574 0.8687574
[776] 0.8687574 0.8687574 0.8687574 0.8687574 0.7557290
[781] 0.7557290 0.7557290 0.7557290 0.7557290 0.7557290
[786] 0.7557290 0.7557290 0.7557290 0.7557290 0.7557290
[791] 0.8759795 0.8759795 0.8759795 0.8759795 0.8759795
[796] 0.8759795 0.8759795 0.8758833 0.8758833 0.8758833
[801] 0.8758833 0.8758833 0.8758833 0.8758833 0.8758833
[806] 0.8758833 0.8758833 0.8758833 0.8758833 0.8758833
[811] 0.8758833 0.8758833 0.8758833 0.8758833 0.7556360
[816] 0.7556360 0.8657795 0.8657795 0.8657795 0.8657795
[821] 0.8657795 0.8657795 0.8657795 0.8657795 0.8657795
[826] 0.8657795 0.8657795 0.8657795 0.8657795 0.8657795
[831] 0.8657795 0.9295921 0.9295921 0.9295921 0.9295921
[836] 0.9295921 0.9295921 0.9295921 0.9295921 0.9295921
[841] 0.9295921 0.7998670 0.7998670 0.7998670 0.7998670
[846] 0.7998670 0.7998670 0.7998670 0.7998670 0.7998670
[851] 0.7998670 0.7998670 0.7998670 0.7998670 0.7998670
[856] 0.7998670 0.7998670 0.8750439 0.8750439 0.8750439
[861] 0.8750439 0.8750439 0.8750439 0.8750439 0.7850714
[866] 0.7850714 0.8862135 0.8862135 0.8862135 0.8862135
[871] 0.8862135 0.8862135 0.8862135 0.8862135 0.8862135
[876] 0.8862135 0.8862135 0.8862135 0.8862135 0.8862135
[881] 0.8862135 0.8862135 0.8862135 0.8862135 0.8862135
[886] 0.8862135 0.8862135 0.8862135 0.8862135 0.8862135
[891] 0.8862135 0.8862135 0.8862135 0.8862135 0.8862135
[896] 0.8862135 0.8862135 0.8466130 0.8466130 0.8466130
[901] 0.8466130 0.8466130 0.8466130 0.8466130 0.8466130
[906] 0.8466130 0.8466130 0.8466130 0.9062867 0.9062867
[911] 0.9062867 0.9062867 0.9062867 0.9062867 0.9062867
[916] 0.9062867 0.9062867 0.9062867 0.9062867 0.9062867
[921] 0.9062867 0.9062867 0.9062867 0.9062867 0.9062867
[926] 0.7358990 0.7358990 0.7358990 0.7358990 0.7358990
[931] 0.7358990 0.7358990 0.7358990 0.7358990 0.8303107
[936] 0.8303107 0.8303107 0.8303107 0.8303107 0.8303107
[941] 0.8303107 0.7462151 0.7462151 0.7462151 0.7462151
[946] 0.7462151 0.7462151 0.7462151 0.7462151 0.7462151
[951] 0.7462151 0.7462151 0.7462151 0.7462151 0.7462151
[956] 0.7462151 0.7462151 0.7462151 0.7462151 0.7462151
[961] 0.7703681 0.7383606 0.7383606 0.7383606 0.7383606
```

```
 [966] 0.7383606 0.7383606 0.7383606 0.7383606 0.7383606
 [971] 0.7383606 0.7383606 0.7383606 0.7383606 0.7383606
 [976] 0.7222720 0.7222720 0.7222720 0.7222720 0.7222720
 [981] 0.7222720 0.7222720 0.7222720 0.7222720 0.7222720
 [986] 0.7222720 0.7222720 0.7222720 0.7222720 0.8206542
 [991] 0.8206542 0.8206542 0.8206542 0.8206542 0.8206542
 [996] 0.8206542 0.8206542 0.7014610 0.7014610 0.7014610
[1001] 0.7014610 0.7014610 0.7014610 0.7014610 0.7014610
[1006] 0.7014610 0.7014610 0.7014610 0.7014610 0.7014610
[1011] 0.7014610 0.8835031 0.8835031 0.8835031 0.8835031
[1016] 0.7227082 0.7227082 0.7227082 0.7227082 0.7227082
[1021] 0.7227082 0.7227082 0.7227082 0.7227082 0.7227082
[1026] 0.7227082 0.7227082 0.7227082 0.7227082 0.7227082
[1031] 0.7227082 0.7227082 0.7227082 0.7227082 0.7227082
[1036] 0.7227082 0.7227082 0.7227082 0.7227082 0.7227082
[1041] 0.7227082 0.7227082 0.7227082 0.7227082 0.7227082
[1046] 0.7227082 0.7227082 0.7227082 0.7227082 0.7227082
[1051] 0.7227082 0.7227082 0.7227082 0.7227082 0.7227082
[1056] 0.7227082 0.7227082 0.7227082 0.7227082 0.8134901
[1061] 0.8134901 0.8134901 0.8134901 0.8134901 0.8134901
[1066] 0.8134901 0.8382973 0.8382973 0.8382973 0.8382973
[1071] 0.8382973 0.8382973 0.8382973 0.8382973 0.8382973
[1076] 0.8382973 0.8382973 0.8382973 0.8382973 0.8382973
[1081] 0.8382973 0.8382973 0.8382973 0.8382973 0.8411904
[1086] 0.8411904 0.8411904 0.7878007 0.7878007 0.7878007
[1091] 0.7878007 0.7878007 0.7878007 0.7878007 0.7878007
[1096] 0.7878007 0.7878007 0.8121770 0.8121770 0.8121770
[1101] 0.8121770 0.8121770 0.8121770 0.8121770 0.8601965
[1106] 0.8601965 0.8601965 0.8601965 0.6780176 0.6780176
[1111] 0.6780176 0.6780176 0.6780176 0.6780176 0.6780176
[1116] 0.7339130 0.7339130 0.7339130 0.7339130 0.7339130
[1121] 0.7339130 0.7339130 0.7339130 0.7339130 0.7339130
[1126] 0.7339130 0.7339130 0.7339130 0.7339130 0.8108317
[1131] 0.8108317 0.8108317 0.8108317 0.8108317 0.8108317
[1136] 0.8108317 0.8108317 0.8108317 0.8108317 0.8108317
[1141] 0.8108317 0.8108317 0.8108317 0.7593922 0.7593922
[1146] 0.7593922 0.7593922 0.7593922 0.7593922 0.7593922
[1151] 0.7593922 0.7593922 0.7593922 0.8670662 0.8670662
[1156] 0.8670662 0.8670662 0.8670662 0.8670662 0.8670662
[1161] 0.8670662 0.8670662 0.8670662 0.8670662 0.8670662
[1166] 0.8670662 0.8670662 0.8315377 0.8315377 0.8315377
[1171] 0.8315377 0.7951299 0.7951299 0.7951299 0.7951299
[1176] 0.7951299 0.7951299 0.7951299 0.7951299 0.7951299
[1181] 0.7951299 0.6735224 0.6735224 0.6735224 0.6735224
[1186] 0.6735224 0.6735224 0.6735224 0.6735224 0.6735224
[1191] 0.6735224 0.6735224 0.6735224 0.6735224 0.6735224
[1196] 0.6735224 0.6735224 0.7921119 0.7921119 0.7921119
[1201] 0.7921119 0.7921119 0.7921119 0.7921119 0.7921119
[1206] 0.7921119 0.7921119 0.7921119 0.7921119 0.7921119
[1211] 0.7921119 0.7921119 0.7921119 0.7921119 0.7921119
[1216] 0.7921119 0.7921119 0.7921119 0.7921119 0.7921119
[1221] 0.7921119 0.8075426 0.8075426 0.8075426 0.8075426
[1226] 0.8075426 0.8075426 0.8075426 0.8075426 0.8205015
[1231] 0.8205015 0.8205015 0.8205015 0.8205015 0.8205015
```

```
[1236] 0.8205015 0.8205015 0.8205015 0.8205015 0.8205015
[1241] 0.8205015 0.8205015 0.8205015 0.8205015 0.8205015
[1246] 0.8205015 0.8205015 0.8205015 0.8205015 0.8205015
[1251] 0.8205015 0.7480169 0.7480169 0.7480169 0.7480169
[1256] 0.7480169 0.7480169 0.7480169 0.7480169 0.7480169
[1261] 0.7480169 0.7480169 0.7480169 0.8199580 0.8199580
[1266] 0.8199580 0.8199580 0.7111585 0.7111585 0.7731372
[1271] 0.7731372 0.7731372 0.7731372 0.7731372 0.7731372
[1276] 0.7731372 0.7731372 0.7731372 0.7731372 0.7731372
[1281] 0.7731372 0.7731372 0.7731372 0.7731372 0.7731372
[1286] 0.7731372 0.7731372 0.7731372 0.7731372 0.7731372
[1291] 0.7731372 0.7731372 0.7731372 0.7731372 0.7731372
[1296] 0.7731372 0.7731372 0.7731372 0.7731372 0.7731372
[1301] 0.7731372 0.7731372 0.7731372 0.7731372 0.8129312
[1306] 0.8129312 0.8129312 0.8129312 0.8129312 0.8129312
[1311] 0.8129312 0.8129312 0.8129312 0.8129312 0.8129312
[1316] 0.8129312 0.8129312 0.8129312 0.8129312 0.8129312
[1321] 0.8129312 0.8129312 0.8129312 0.8129312 0.8129312
[1326] 0.8129312 0.7316860 0.7316860 0.7316860 0.7316860
[1331] 0.7316860 0.7316860 0.7316860 0.8248843 0.6878304
[1336] 0.6878304 0.5480240 0.5480240 0.5480240 0.5480240
[1341] 0.5480240 0.5480240 0.5480240 0.8179600 0.8179600
[1346] 0.8179600 0.8179600 0.8179600 0.8179600 0.8179600
[1351] 0.8179600 0.8179600 0.8179600 0.8179600 0.8179600
[1356] 0.8179600 0.8179600 0.8179600 0.8179600 0.8179600
[1361] 0.8179600 0.8179600 0.8179600 0.8179600 0.8179600
[1366] 0.8179600 0.8179600 0.8179600 0.8179600 0.8179600
[1371] 0.8179600 0.8179600 0.8179600 0.8392713 0.8392713
[1376] 0.8392713 0.8392713 0.8392713 0.8392713 0.8392713
[1381] 0.8392713 0.8392713 0.7689269 0.7689269 0.7689269
[1386] 0.7689269 0.7689269 0.7689269 0.7689269 0.7689269
[1391] 0.7689269 0.8202807 0.8202807 0.8202807 0.8202807
[1396] 0.8202807 0.8202807 0.7014544 0.7014544 0.7014544
[1401] 0.7800127 0.7800127 0.7800127 0.7800127 0.7800127
[1406] 0.7800127 0.7800127 0.7800127 0.7800127 0.7800127
[1411] 0.7800127 0.7800127 0.7800127 0.7800127 0.7800127
[1416] 0.6318107 0.6318107 0.6318107 0.6318107 0.6318107
[1421] 0.6318107 0.6318107 0.6318107 0.6318107 0.6318107
[1426] 0.8568135 0.8568135 0.8568135 0.8568135 0.8568135
[1431] 0.8568135 0.8568135 0.8568135 0.8568135 0.8568135
[1436] 0.8568135 0.8568135 0.8568135 0.8568135 0.8568135
[1441] 0.8568135 0.8568135 0.8568135 0.8568135 0.8568135
[1446] 0.8568135 0.8568135 0.8568135 0.8568135 0.8568135
[1451] 0.8568135 0.8568135 0.8568135 0.8568135 0.8568135
[1456] 0.8568135 0.8568135 0.8568135 0.8568135 0.8568135
[1461] 0.8870107 0.8870107 0.8870107 0.8870107 0.8870107
[1466] 0.8870107 0.8870107 0.8870107 0.8870107 0.8514735
[1471] 0.8514735 0.8514735 0.8514735 0.8514735 0.8514735
[1476] 0.8514735 0.8514735 0.8514735 0.7867862 0.7867862
[1481] 0.7867862 0.7867862 0.7867862 0.7867862 0.7867862
[1486] 0.7863844 0.7863844 0.7863844 0.7863844 0.7863844
[1491] 0.6704393 0.6704393 0.6704393 0.6704393 0.6704393
[1496] 0.6704393 0.6704393 0.6704393 0.8588958 0.8588958
[1501] 0.8588958 0.8588958 0.8588958 0.8588958 0.8588958
```

```
[1506] 0.8588958 0.8588958 0.8588958 0.8030871 0.8030871
[1511] 0.8030871 0.8030871 0.8030871 0.8030871 0.8030871
[1516] 0.8030871 0.8030871 0.7733453 0.7733453 0.7733453
[1521] 0.7160974 0.7160974 0.7160974 0.7160974 0.7160974
[1526] 0.7160974 0.7160974 0.7160974 0.7160974 0.7160974
[1531] 0.7160974 0.7160974 0.7973947 0.7973947 0.7973947
[1536] 0.7595786 0.7595786 0.7595786 0.7595786 0.7595786
[1541] 0.7595786 0.7595786 0.7595786 0.7595786 0.7595786
[1546] 0.7910362 0.7910362 0.7910362 0.7910362 0.7316392
[1551] 0.7316392 0.7316392 0.7316392 0.7316392 0.7316392
[1556] 0.8970650 0.8970650 0.8970650 0.8970650 0.8970650
[1561] 0.8970650 0.8970650 0.8970650 0.8970650 0.8970650
[1566] 0.8970650 0.8970650 0.8970650 0.8970650 0.8970650
[1571] 0.8970650 0.8970650 0.8970650 0.8970650 0.8970650
[1576] 0.8970650 0.8970650 0.8970650 0.8970650 0.8970650
[1581] 0.8970650 0.8151148 0.8151148 0.8151148 0.8151148
[1586] 0.8151148 0.8151148 0.8151148 0.8151148 0.8151148
[1591] 0.8151148 0.8151148 0.8151148 0.8151148 0.8151148
[1596] 0.8151148 0.8151148 0.8151148 0.8151148 0.8151148
[1601] 0.8151148 0.8151148 0.8151148 0.8151148 0.8151148
[1606] 0.8151148 0.8721354 0.8721354 0.6104764 0.6104764
[1611] 0.6104764 0.6104764 0.6104764 0.6104764 0.7552623
[1616] 0.7552623 0.7552623 0.7552623 0.7552623 0.7552623
[1621] 0.7552623 0.7552623 0.7552623 0.7552623 0.7552623
[1626] 0.7552623 0.7552623 0.7552623 0.7552623 0.7384571
[1631] 0.7384571 0.7384571 0.7384571 0.7384571 0.7384571
[1636] 0.7384571 0.7384571 0.7384571 0.7384571 0.7384571
[1641] 0.7384571 0.7384571 0.7384571 0.6940212 0.6940212
[1646] 0.6940212 0.6940212 0.8728949 0.8728949 0.8728949
[1651] 0.8728949 0.8728949 0.8728949 0.8728949 0.8728949
[1656] 0.8728949 0.6774428 0.6774428 0.8077589 0.8384489
[1661] 0.8384489 0.8384489 0.8384489 0.7989000 0.7989000
[1666] 0.7989000 0.7989000 0.7989000 0.7989000 0.7989000
[1671] 0.7989000 0.7227322 0.7227322 0.7227322 0.7227322
[1676] 0.7227322 0.7227322 0.7227322 0.7227322 0.7227322
[1681] 0.7227322 0.7227322 0.7227322 0.7227322 0.7227322
[1686] 0.7227322 0.7227322 0.7227322 0.7227322 0.7227322
[1691] 0.7227322 0.7227322 0.7227322 0.7227322 0.7227322
[1696] 0.7227322 0.7227322 0.7227322 0.7227322 0.7227322
[1701] 0.7227322 0.8435158 0.8435158 0.8435158 0.8435158
[1706] 0.8435158 0.8435158 0.8435158 0.8435158 0.8435158
[1711] 0.8435158 0.8435158 0.8435158 0.8435158 0.8435158
[1716] 0.8435158 0.8435158 0.8435158 0.8435158 0.8435158
[1721] 0.8435158 0.8435158 0.8435158 0.8435158 0.8435158
[1726] 0.7748621 0.7748621 0.7748621 0.7748621 0.7748621
[1731] 0.7748621 0.7748621 0.7748621 0.7748621 0.7748621
[1736] 0.7748621 0.7748621 0.6775068 0.6775068 0.6485148
[1741] 0.6485148 0.6485148 0.6485148 0.6485148 0.6485148
[1746] 0.8810513 0.8810513 0.8810513 0.8810513 0.8810513
[1751] 0.8810513 0.8810513 0.8810513 0.8810513 0.8810513
[1756] 0.8810513 0.7101959 0.7101959 0.7101959 0.7101959
[1761] 0.7101959 0.7101959 0.7101959 0.7101959 0.7101959
[1766] 0.7101959 0.7101959 0.7101959 0.7101959 0.7101959
[1771] 0.6048006 0.6048006 0.6048006 0.6048006 0.6048006
```

```
[1776] 0.6048006 0.6048006 0.6048006 0.6048006 0.6048006
[1781] 0.6048006 0.7991581 0.7991581 0.7991581 0.7991581
[1786] 0.7991581 0.7991581 0.7991581 0.7991581 0.7991581
[1791] 0.7991581 0.7991581 0.7991581 0.7991581 0.7991581
[1796] 0.7991581 0.7991581 0.7991581 0.7991581 0.7991581
[1801] 0.7991581 0.8522538 0.8522538 0.8522538 0.8522538
[1806] 0.8522538 0.8522538 0.8522538 0.8522538 0.8522538
[1811] 0.8522538 0.8522538 0.8522538 0.8522538 0.8522538
[1816] 0.8522538 0.8522538 0.8522538 0.8522538 0.8522538
[1821] 0.8522538 0.8522538 0.8522538 0.8522538 0.8522538
[1826] 0.8522538 0.8522538 0.8522538 0.8522538 0.8522538
[1831] 0.8522538 0.8522538 0.8522538 0.8522538 0.8522538
[1836] 0.8522538 0.8522538 0.8522538 0.8522538 0.8522538
[1841] 0.8522538 0.8522538 0.7477716 0.7477716 0.7477716
[1846] 0.7477716 0.7477716 0.7477716 0.7477716 0.7477716
[1851] 0.7477716 0.7477716 0.7477716 0.7477716 0.7477716
[1856] 0.7477716 0.7477716 0.7477716 0.7008008 0.7008008
[1861] 0.7008008 0.7008008 0.8157709 0.8157709 0.8157709
[1866] 0.8157709 0.8157709 0.8157709 0.8157709 0.8157709
[1871] 0.8157709 0.8157709 0.8157709 0.8157709 0.8157709
[1876] 0.8157709 0.8157709 0.8990674 0.8990674 0.8990674
[1881] 0.8990674 0.8990674 0.8990674 0.8990674 0.8990674
[1886] 0.8990674 0.8990674 0.8990674 0.8990674 0.8990674
[1891] 0.8990674 0.8990674 0.8990674 0.8990674 0.8990674
[1896] 0.8792765 0.8792765 0.8792765 0.8792765 0.8225867
[1901] 0.8225867 0.8225867 0.8225867 0.8225867 0.8225867
[1906] 0.8225867 0.8225867 0.8225867 0.8225867 0.8225867
[1911] 0.8225867 0.8017823 0.8017823 0.8017823 0.8017823
[1916] 0.8017823 0.8017823 0.8017823 0.8017823 0.8017823
[1921] 0.8017823 0.8628983 0.8628983 0.8574278 0.8574278
[1926] 0.8574278 0.8574278 0.8574278 0.8574278 0.8574278
[1931] 0.8574278 0.8574278 0.8574278 0.8790040 0.8790040
[1936] 0.8790040 0.8790040 0.8790040 0.8790040 0.8790040
[1941] 0.8790040 0.8790040 0.8790040 0.8790040 0.8790040
[1946] 0.7757391 0.7757391 0.7757391 0.7757391 0.8777224
[1951] 0.8777224 0.8777224 0.8777224 0.8777224 0.8777224
[1956] 0.8777224 0.8777224 0.8777224 0.8777224 0.8777224
[1961] 0.8777224 0.8777224 0.8777224 0.7822744 0.7822744
[1966] 0.7822744 0.7822744 0.7822744 0.7822744 0.7822744
[1971] 0.7822744 0.7822744 0.7822744 0.8387775 0.8356654
[1976] 0.8356654 0.8356654 0.8356654 0.8356654 0.8356654
[1981] 0.8356654 0.8488983 0.8488983 0.8488983 0.8488983
[1986] 0.8488983 0.8488983 0.8488983 0.8488983 0.8488983
[1991] 0.6352013 0.6352013 0.6352013 0.6352013 0.6352013
[1996] 0.6352013 0.6352013 0.6352013 0.6352013 0.6352013
[2001] 0.6352013 0.6352013 0.6352013 0.6352013 0.7621640
[2006] 0.7621640 0.7621640 0.7621640 0.7621640 0.7621640
[2011] 0.7621640 0.7621640 0.7621640 0.7621640 0.7621640
[2016] 0.7621640 0.7621640 0.7621640 0.7621640 0.7621640
[2021] 0.7621640 0.7621640 0.7621640 0.7966061 0.7966061
[2026] 0.7966061 0.7966061 0.7966061 0.7966061 0.6321321
[2031] 0.6321321 0.6321321 0.6321321 0.6321321 0.6321321
[2036] 0.7881741 0.7881741 0.7881741 0.7881741 0.7881741
[2041] 0.7881741 0.7881741 0.7881741 0.7881741 0.6879246
```

```
[2046] 0.6879246 0.6879246 0.6879246 0.6879246 0.6879246
[2051] 0.6879246 0.6879246 0.6879246 0.6879246 0.6879246
[2056] 0.6879246 0.6879246 0.6879246 0.6879246 0.6879246
[2061] 0.6879246 0.6879246 0.6879246 0.6879246 0.8219778
[2066] 0.8219778 0.8219778 0.8219778 0.8219778 0.8219778
[2071] 0.8219778 0.8219778 0.8219778 0.8219778 0.8219778
[2076] 0.8219778 0.8219778 0.8219778 0.8219778 0.8219778
[2081] 0.8219778 0.8219778 0.8219778 0.7920121 0.7920121
[2086] 0.7920121 0.7920121 0.7920121 0.7920121 0.7920121
```

```
(y <- rbinom(sum(persons), 1, prob))
```

```
   [1] 0 1 1 1 0 1 0 1 0 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1
  [28] 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1
  [55] 1 1 1 1 1 0 1 0 1 1 1 1 0 1 0 1 1 0 1 1 0 1 1 1 1 1 1
  [82] 1 1 1 1 1 1 1 1 0 1 0 1 0 1 0 1 1 1 1 1 1 1 1 0 1 1 0
 [109] 1 1 1 0 0 1 0 1 0 0 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1
 [136] 1 1 1 1 1 1 1 1 1 0 1 1 1 1 0 0 1 1 1 1 1 0 1 1 0 1 0
 [163] 0 0 0 1 0 0 0 1 0 1 1 0 1 1 0 0 1 0 0 1 1 1 0 0 1 1 1
 [190] 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 [217] 0 1 1 1 1 1 0 1 1 0 1 1 0 1 1 1 1 0 0 1 1 1 1 1 0 1 1
 [244] 0 0 1 0 1 1 1 1 0 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 [271] 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1
 [298] 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 0 0 1 1 1 0
 [325] 1 0 0 0 1 0 0 0 1 1 1 1 1 1 1 1 1 0 1 0 0 1 1 1 0 1 0
 [352] 1 1 1 1 1 1 0 1 0 1 1 1 1 1 1 0 1 1 1 1 1 1 0 1 1 1 0
 [379] 0 0 0 1 1 1 1 1 1 0 0 1 1 1 0 1 1 1 1 0 1 0 1 1 0 1 1
 [406] 1 1 1 1 1 1 1 1 1 0 1 1 0 0 1 1 1 0 0 1 0 1 1 1 1 1 0
 [433] 0 1 0 0 0 1 0 1 0 1 1 0 1 1 1 1 1 0 0 1 1 0 1 1 1 1 1
 [460] 1 1 1 1 1 1 1 1 1 1 1 0 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0
 [487] 1 1 1 1 0 1 0 1 1 1 1 1 1 1 1 1 1 1 0 1 1 0 1 0 1 0 1
 [514] 1 0 1 0 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1
 [541] 0 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 0 1 1 1 1
 [568] 1 1 1 1 1 1 1 1 0 1 1 0 1 1 0 1 1 1 1 1 0 1 1 1 1 1 1
 [595] 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 0 1 1 1 1 1 1 1
 [622] 1 1 1 1 1 1 1 1 0 1 1 0 1 1 1 1 1 1 1 1 0 1 1 0 1 1 1
 [649] 1 1 1 1 0 1 1 0 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1
 [676] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1
 [703] 1 1 0 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 0
 [730] 0 0 1 1 1 1 1 0 1 0 1 1 1 1 1 1 0 1 0 1 1 1 1 0 0 1 0
 [757] 1 1 0 0 1 1 0 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 0 1 1
 [784] 1 1 1 1 0 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1
 [811] 1 1 1 0 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 0 1
 [838] 1 1 1 1 0 1 1 0 1 1 1 1 0 1 0 1 1 1 1 1 1 1 1 1 1 1 1
 [865] 0 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1
 [892] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1
 [919] 1 1 0 1 1 0 1 1 1 0 0 0 1 0 1 1 0 0 1 1 1 1 1 1 1 0 1
 [946] 0 1 1 1 1 1 0 1 1 1 1 1 1 0 0 0 1 1 1 1 0 1 1 1 1 1 1
 [973] 1 1 1 0 1 1 0 1 1 0 1 0 1 0 1 1 0 1 1 1 0 1 1 1 1 1 1
[1000] 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 0 0 0 1 1
[1027] 0 0 1 1 1 1 1 1 0 0 1 0 1 1 1 1 1 1 1 1 1 0 1 1 1 1 0
[1054] 1 0 1 1 1 1 1 0 1 1 0 1 1 0 1 0 1 1 1 1 1 1 1 1 1 0 0
[1081] 1 1 1 0 1 1 1 0 0 1 1 1 1 1 0 1 1 1 0 1 1 1 1 1 1 1 1
[1108] 1 1 1 1 0 1 0 0 1 1 0 1 1 1 1 1 0 1 1 1 1 1 0 1 1 1 1
[1135] 1 1 1 1 0 0 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```
[1162] 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 0 1 0 0 1 1 1 0 1 0 1
[1189] 1 1 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 0 0
[1216] 1 1 1 0 1 1 1 0 0 0 1 1 1 1 0 1 0 1 1 1 1 1 1 0 1 1 1
[1243] 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1
[1270] 1 0 0 1 1 1 0 1 1 1 0 1 0 0 1 1 1 1 0 1 1 1 1 1 1 1 1
[1297] 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1
[1324] 1 1 0 1 1 0 0 0 1 1 1 0 0 0 1 1 0 1 0 1 1 1 1 1 1 1 1
[1351] 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 0 0 1 1 1 0 1 0 1 1 1
[1378] 1 1 0 1 1 1 0 1 1 1 1 1 0 0 0 1 1 0 1 1 1 1 0 1 1 1 1
[1405] 0 1 1 0 0 0 1 1 1 1 1 0 0 1 0 1 0 0 0 0 1 1 1 1 0 1 1
[1432] 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
[1459] 1 0 1 1 1 1 1 1 1 0 1 1 0 1 1 0 1 1 1 1 0 1 1 1 1 0 1
[1486] 1 1 1 1 1 1 0 1 0 0 1 0 0 0 1 1 0 1 1 1 1 1 0 1 1 1 1
[1513] 1 1 1 1 1 0 1 1 0 1 0 0 1 1 0 0 1 1 0 1 1 1 0 1 1 1 0
[1540] 1 1 0 1 1 0 1 1 1 1 0 1 0 1 1 1 1 1 1 1 1 1 1 1 0 1 1
[1567] 0 1 1 0 1 1 1 1 1 0 1 1 1 1 1 1 1 1 0 1 1 1 1 1 0 1 1
[1594] 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 0 1 0 0 0 0 1 1 1 1 1 1
[1621] 1 1 1 1 1 0 1 1 1 0 1 1 1 1 1 1 0 1 1 0 1 0 1 1 0 1 0
[1648] 1 1 1 0 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 0 0 1 1 1
[1675] 1 1 0 1 0 1 0 1 1 1 1 0 1 1 1 1 0 1 0 0 0 1 1 1 1 1 0
[1702] 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1 1 1 1 1 1 1 0 1 1 1 1
[1729] 1 1 1 1 1 0 1 1 0 1 1 0 1 0 0 0 0 1 1 1 1 1 1 0 1 1 1
[1756] 1 1 1 1 0 1 1 0 1 0 1 1 1 1 1 0 1 1 1 0 0 1 1 0 0 1 1
[1783] 1 1 1 0 1 1 1 1 0 1 0 1 1 1 1 1 1 0 1 0 1 1 1 1 1 0 1
[1810] 0 1 1 1 0 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 0 1 1 1 1
[1837] 1 1 1 1 1 1 1 1 1 1 0 1 1 0 1 1 1 1 1 1 1 0 1 1 0 1 1
[1864] 0 1 0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 1 1 1
[1891] 1 1 1 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1
[1918] 1 0 1 0 1 1 1 1 1 0 0 1 1 0 1 1 1 1 1 1 1 0 1 1 0 1 1
[1945] 1 1 1 1 1 0 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 0 1 0
[1972] 1 0 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 0 1 0 1 1 1 1 0 1 0
[1999] 1 1 0 1 1 1 0 1 1 1 1 1 1 0 1 1 1 1 0 1 0 1 1 1 0 1 1
[2026] 0 0 1 1 0 1 0 0 1 0 1 0 1 1 1 1 1 0 1 1 1 0 0 1 1 0 1
[2053] 1 1 0 1 1 0 1 0 1 0 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1
[2080] 0 1 1 1 1 1 1 1 1 1 1
```

```r
# create
d <- as.data.frame(cbind(y, d))
d$country <- as.factor(d$country)
d$site <- as.factor(d$site)
```

Frequentist random effects model

```r
require(lme4)
fit0 = glmer(data = d, formula = y ~ (1 | country/site), family = binomial)
print(fit0)
```

```
Generalized linear mixed model fit by maximum likelihood
  (Laplace Approximation) [glmerMod]
 Family: binomial  ( logit )
Formula: y ~ (1 | country/site)
   Data: d
      AIC       BIC    logLik  deviance  df.resid
 2142.788  2159.723 -1068.394  2136.788      2087
```

```
Random effects:
 Groups       Name        Std.Dev.
 site:country (Intercept) 0.5092
 country      (Intercept) 0.1164
Number of obs: 2090, groups:
site:country, 185; country, 25
Fixed Effects:
(Intercept)
      1.375
```

```
# (fit0fci <- confint(fit0)) # takes some time

coef(fit0)
```

```
$`site:country`
       (Intercept)
1:1      1.2843616
2:1      1.2389101
3:1      1.5427703
4:2      1.6932682
5:2      1.6043910
6:2      1.4242760
7:2      1.4260246
8:2      1.2348317
9:2      1.1199178
10:2     1.7336906
11:2     1.3077852
12:2     1.1220384
13:2     0.9986295
14:2     1.6353192
15:2     1.5980762
16:2     1.6353192
17:2     0.6657858
18:2     1.1012165
19:2     1.7336906
20:2     1.4242760
21:2     1.0703280
22:2     1.2859965
23:2     1.1188143
24:3     1.4710669
25:4     1.2467380
26:4     1.6861931
27:4     1.5515094
28:4     1.7162749
29:4     1.1555860
30:4     1.4650407
31:4     1.6951250
32:5     0.5479463
33:5     1.7156535
34:5     1.0269988
35:5     1.3654717
36:5     1.5887700
37:5     1.2947098
38:5     1.0581983
39:5     1.0685164
```

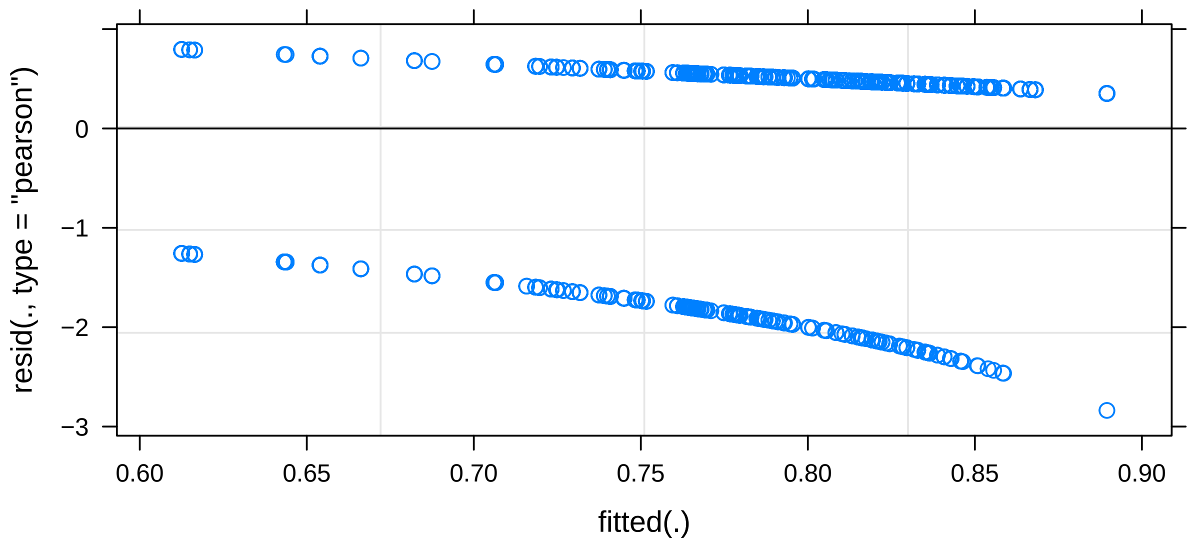
40:5	1.4210966
41:5	1.2680890
42:5	0.5580186
43:5	1.5786491
44:5	1.3446504
45:5	1.5296684
46:5	1.0328302
47:5	1.1360107
48:5	1.7680657
49:5	1.1826960
50:5	1.3453215
51:5	1.1826960
52:5	1.3654717
53:5	1.5242808
54:5	1.5114017
55:5	1.3654717
56:5	1.8636366
57:6	1.3225067
58:6	1.7142442
59:6	1.5990136
60:6	1.1960405
61:6	1.7385546
62:6	1.8191583
63:6	1.6365245
64:6	1.6511100
65:6	1.2765077
66:6	1.0926192
67:6	1.3225067
68:7	1.1902771
69:8	1.0111499
70:9	1.6573119
71:9	1.3886327
72:9	1.4402482
73:9	1.5665901
74:9	1.4655301
75:9	1.6833029
76:9	1.5419903
77:9	1.2223810
78:9	1.6478017
79:9	1.2256470
80:9	2.0032502
81:9	1.7634282
82:9	1.4073805
83:9	0.9501040
84:9	1.2383441
85:10	1.2124776
86:10	1.1766765
87:10	1.6855756
88:10	0.8841717
89:10	1.4965275
90:10	1.6855756
91:11	1.5564605
92:11	1.1095450
93:11	1.2597538

94:11	1.1811501
95:11	1.5156477
96:11	1.2005005
97:11	1.4597484
98:11	1.5564605
99:12	1.0598973
100:12	1.5079158
101:12	1.3420743
102:12	1.5610109
103:12	1.8591784
104:12	1.3302722
105:12	1.1943427
106:12	1.2519149
107:12	1.4804126
108:13	1.1118447
109:13	1.3020171
110:13	1.6278278
111:14	1.3437597
112:14	1.4773711
113:14	1.3263917
114:14	1.6109573
115:14	1.0834903
116:14	1.4280495
117:14	1.0033991
118:14	1.0834903
119:14	1.6626597
120:14	1.3625403
121:14	1.1795561
122:14	1.2300416
123:14	1.2934624
124:14	1.2549274
125:14	0.5540992
126:14	1.8817242
127:14	1.5510539
128:14	1.3625403
129:14	1.2770027
130:14	1.6060410
131:14	0.7705959
132:15	1.2117777
133:15	1.7370846
134:15	1.2891580
135:15	0.8076574
136:15	1.2891580
137:15	1.2117777
138:15	1.5611263
139:15	1.2221409
140:15	1.5554766
141:15	1.6657772
142:15	1.4746622
143:15	0.6331758
144:16	1.7128009
145:16	1.1889732
146:16	1.1136287
147:16	1.5321195

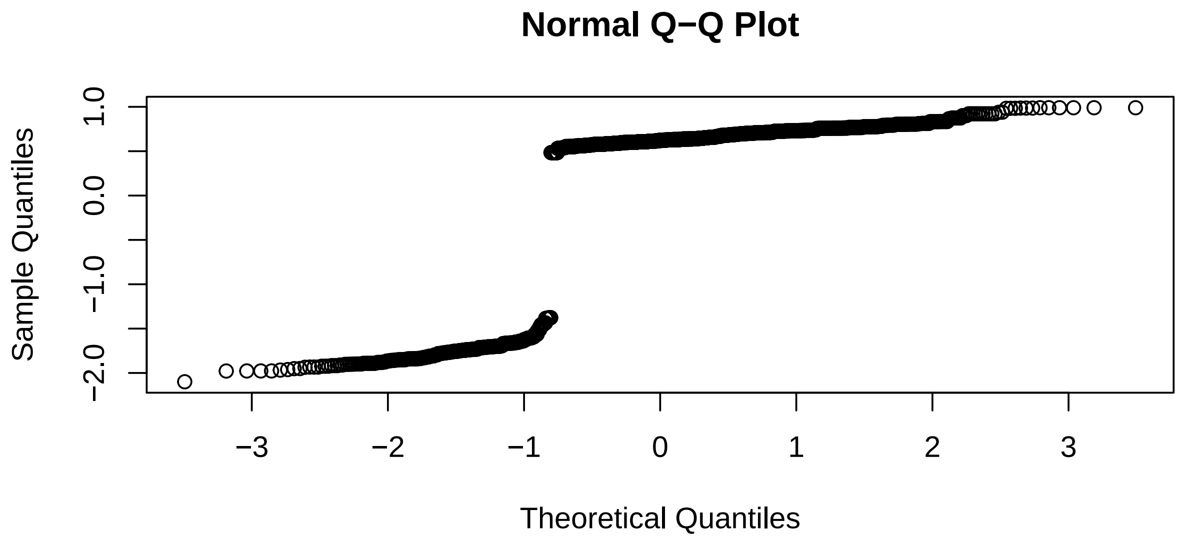
```
148:16   1.4718972
149:16   1.4251256
150:16   1.3328437
151:16   1.3018396
152:16   1.0567703
153:16   1.4889184
154:17   1.4612359
155:17   1.4744058
156:17   0.6322723
157:17   1.6075938
158:17   1.3613577
159:17   0.9165517
160:17   1.3982735
161:18   1.5953803
162:18   1.4062106
163:18   1.3303291
164:19   1.5385907
165:20   1.6191446
166:20   1.3299881
167:20   1.4438375
168:20   1.3741554
169:20   1.4704728
170:20   1.1937280
171:20   1.4438375
172:21   1.5541729
173:22   1.5125445
174:22   1.1985265
175:23   1.4247766
176:24   1.6630293
177:24   1.3392737
178:24   1.1851666
179:25   1.2262878
180:25   1.2202689
181:25   0.8236081
182:25   1.3499900
183:25   0.9952681
184:25   1.6639388
185:25   1.6706012

$country
   (Intercept)
1     1.371584
2     1.345422
3     1.379644
4     1.421335
5     1.284678
6     1.439830
7     1.364970
8     1.355610
9     1.457323
10    1.369032
11    1.366369
12    1.385870
13    1.370312
```

```
14     1.294986
15     1.330901
16     1.368274
17     1.334336
18     1.385479
19     1.383173
20     1.387822
21     1.383987
22     1.372610
23     1.377225
24     1.377930
25     1.339472

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

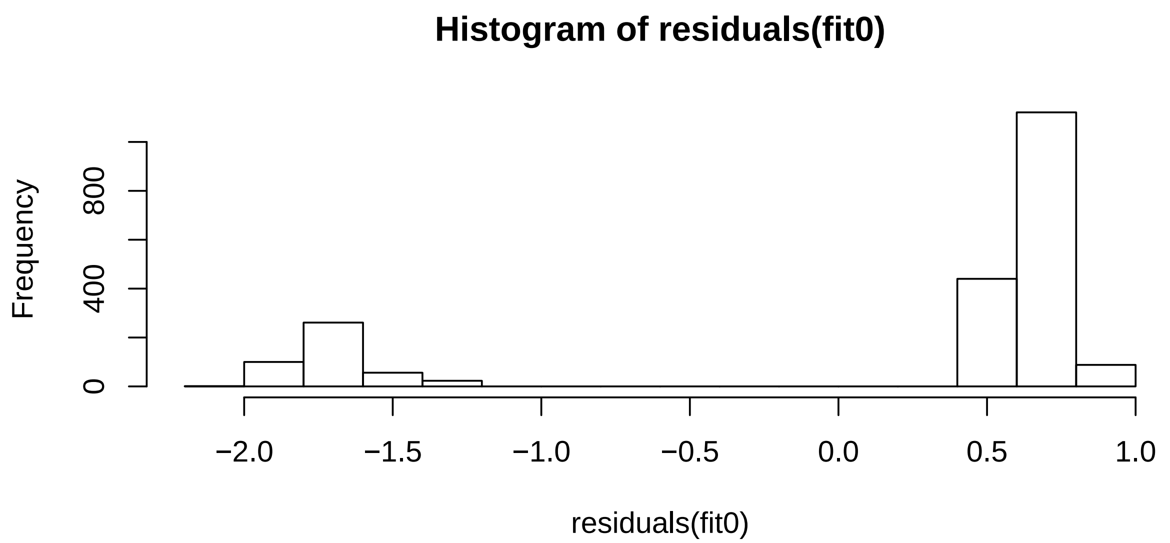
```
# are these sensible from binomial?
plot(fit0)
```




```
qqnorm(residuals(fit0))
```




```r
hist(residuals(fit0))
```




```r
fit0r <- ranef(fit0, condVar = TRUE)  # frequentist estimates of random effects

coef(fit0)
```

```
$`site:country`
       (Intercept)
1:1      1.2843616
2:1      1.2389101
3:1      1.5427703
4:2      1.6932682
5:2      1.6043910
6:2      1.4242760
```

```
7:2       1.4260246
8:2       1.2348317
9:2       1.1199178
10:2      1.7336906
11:2      1.3077852
12:2      1.1220384
13:2      0.9986295
14:2      1.6353192
15:2      1.5980762
16:2      1.6353192
17:2      0.6657858
18:2      1.1012165
19:2      1.7336906
20:2      1.4242760
21:2      1.0703280
22:2      1.2859965
23:2      1.1188143
24:3      1.4710669
25:4      1.2467380
26:4      1.6861931
27:4      1.5515094
28:4      1.7162749
29:4      1.1555860
30:4      1.4650407
31:4      1.6951250
32:5      0.5479463
33:5      1.7156535
34:5      1.0269988
35:5      1.3654717
36:5      1.5887700
37:5      1.2947098
38:5      1.0581983
39:5      1.0685164
40:5      1.4210966
41:5      1.2680890
42:5      0.5580186
43:5      1.5786491
44:5      1.3446504
45:5      1.5296684
46:5      1.0328302
47:5      1.1360107
48:5      1.7680657
49:5      1.1826960
50:5      1.3453215
51:5      1.1826960
52:5      1.3654717
53:5      1.5242808
54:5      1.5114017
55:5      1.3654717
56:5      1.8636366
57:6      1.3225067
58:6      1.7142442
59:6      1.5990136
60:6      1.1960405
```

61:6	1.7385546
62:6	1.8191583
63:6	1.6365245
64:6	1.6511100
65:6	1.2765077
66:6	1.0926192
67:6	1.3225067
68:7	1.1902771
69:8	1.0111499
70:9	1.6573119
71:9	1.3886327
72:9	1.4402482
73:9	1.5665901
74:9	1.4655301
75:9	1.6833029
76:9	1.5419903
77:9	1.2223810
78:9	1.6478017
79:9	1.2256470
80:9	2.0032502
81:9	1.7634282
82:9	1.4073805
83:9	0.9501040
84:9	1.2383441
85:10	1.2124776
86:10	1.1766765
87:10	1.6855756
88:10	0.8841717
89:10	1.4965275
90:10	1.6855756
91:11	1.5564605
92:11	1.1095450
93:11	1.2597538
94:11	1.1811501
95:11	1.5156477
96:11	1.2005005
97:11	1.4597484
98:11	1.5564605
99:12	1.0598973
100:12	1.5079158
101:12	1.3420743
102:12	1.5610109
103:12	1.8591784
104:12	1.3302722
105:12	1.1943427
106:12	1.2519149
107:12	1.4804126
108:13	1.1118447
109:13	1.3020171
110:13	1.6278278
111:14	1.3437597
112:14	1.4773711
113:14	1.3263917
114:14	1.6109573

115:14	1.0834903
116:14	1.4280495
117:14	1.0033991
118:14	1.0834903
119:14	1.6626597
120:14	1.3625403
121:14	1.1795561
122:14	1.2300416
123:14	1.2934624
124:14	1.2549274
125:14	0.5540992
126:14	1.8817242
127:14	1.5510539
128:14	1.3625403
129:14	1.2770027
130:14	1.6060410
131:14	0.7705959
132:15	1.2117777
133:15	1.7370846
134:15	1.2891580
135:15	0.8076574
136:15	1.2891580
137:15	1.2117777
138:15	1.5611263
139:15	1.2221409
140:15	1.5554766
141:15	1.6657772
142:15	1.4746622
143:15	0.6331758
144:16	1.7128009
145:16	1.1889732
146:16	1.1136287
147:16	1.5321195
148:16	1.4718972
149:16	1.4251256
150:16	1.3328437
151:16	1.3018396
152:16	1.0567703
153:16	1.4889184
154:17	1.4612359
155:17	1.4744058
156:17	0.6322723
157:17	1.6075938
158:17	1.3613577
159:17	0.9165517
160:17	1.3982735
161:18	1.5953803
162:18	1.4062106
163:18	1.3303291
164:19	1.5385907
165:20	1.6191446
166:20	1.3299881
167:20	1.4438375
168:20	1.3741554

```
169:20   1.4704728
170:20   1.1937280
171:20   1.4438375
172:21   1.5541729
173:22   1.5125445
174:22   1.1985265
175:23   1.4247766
176:24   1.6630293
177:24   1.3392737
178:24   1.1851666
179:25   1.2262878
180:25   1.2202689
181:25   0.8236081
182:25   1.3499900
183:25   0.9952681
184:25   1.6639388
185:25   1.6706012

$country
   (Intercept)
1     1.371584
2     1.345422
3     1.379644
4     1.421335
5     1.284678
6     1.439830
7     1.364970
8     1.355610
9     1.457323
10    1.369032
11    1.366369
12    1.385870
13    1.370312
14    1.294986
15    1.330901
16    1.368274
17    1.334336
18    1.385479
19    1.383173
20    1.387822
21    1.383987
22    1.372610
23    1.377225
24    1.377930
25    1.339472

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

```r
ranef(fit0, condVar = TRUE)
```

```
$`site:country`
        (Intercept)
1:1    -0.090241500
2:1    -0.135693083
```

```
3:1     0.168167183
4:2     0.318665031
5:2     0.229787836
6:2     0.049672821
7:2     0.051421470
8:2    -0.139771436
9:2    -0.254685346
10:2    0.359087509
11:2   -0.066817967
12:2   -0.252564721
13:2   -0.375973664
14:2    0.260716099
15:2    0.223473052
16:2    0.260716099
17:2   -0.708817308
18:2   -0.273386651
19:2    0.359087509
20:2    0.049672821
21:2   -0.304275151
22:2   -0.088606682
23:2   -0.255788825
24:3    0.096463716
25:4   -0.127865088
26:4    0.311589957
27:4    0.176906285
28:4    0.341671801
29:4   -0.219017184
30:4    0.090437517
31:4    0.320521842
32:5   -0.826656836
33:5    0.341050389
34:5   -0.347604361
35:5   -0.009131459
36:5    0.214166833
37:5   -0.079893292
38:5   -0.316404879
39:5   -0.306086740
40:5    0.046493500
41:5   -0.106514100
42:5   -0.816584522
43:5    0.204045971
44:5   -0.029952717
45:5    0.155065293
46:5   -0.341772915
47:5   -0.238592481
48:5    0.393462545
49:5   -0.191907103
50:5   -0.029281651
51:5   -0.191907103
52:5   -0.009131459
53:5    0.149677688
54:5    0.136798611
55:5   -0.009131459
56:5    0.489033425
```

```
57:6   -0.052096467
58:6    0.339641098
59:6    0.224410425
60:6   -0.178562589
61:6    0.363951492
62:6    0.444555189
63:6    0.261921373
64:6    0.276506863
65:6   -0.098095445
66:6   -0.281983980
67:6   -0.052096467
68:7   -0.184326065
69:8   -0.363453215
70:9    0.282708751
71:9    0.014029544
72:9    0.065645101
73:9    0.191987006
74:9    0.090926925
75:9    0.308699802
76:9    0.167387129
77:9   -0.152222174
78:9    0.273198519
79:9   -0.148956094
80:9    0.628647028
81:9    0.388825066
82:9    0.032777345
83:9   -0.424499109
84:9   -0.136259082
85:10  -0.162125511
86:10  -0.197926680
87:10   0.310972492
88:10  -0.490431404
89:10   0.121924374
90:10   0.310972492
91:11   0.181857380
92:11  -0.265058135
93:11  -0.114849363
94:11  -0.193453066
95:11   0.141044549
96:11  -0.174102592
97:11   0.085145254
98:11   0.181857380
99:12  -0.314705815
100:12  0.133312647
101:12 -0.032528792
102:12  0.186407778
103:12  0.484575261
104:12 -0.044330972
105:12 -0.180260450
106:12 -0.122688281
107:12  0.105809495
108:13 -0.262758401
109:13 -0.072586069
110:13  0.253224715
```

```
111:14 -0.030843389
112:14  0.102767915
113:14 -0.048211456
114:14  0.236354184
115:14 -0.291112853
116:14  0.053446342
117:14 -0.371204051
118:14 -0.291112853
119:14  0.288056591
120:14 -0.012062873
121:14 -0.195046988
122:14 -0.144561501
123:14 -0.081140774
124:14 -0.119675697
125:14 -0.820503948
126:14  0.507121023
127:14  0.176450799
128:14 -0.012062873
129:14 -0.097600408
130:14  0.231437862
131:14 -0.604007262
132:15 -0.162825475
133:15  0.362481467
134:15 -0.085445163
135:15 -0.566945729
136:15 -0.085445163
137:15 -0.162825475
138:15  0.186523140
139:15 -0.152462239
140:15  0.180873448
141:15  0.291174095
142:15  0.100059082
143:15 -0.741427363
144:16  0.338197774
145:16 -0.185629966
146:16 -0.260974426
147:16  0.157516386
148:16  0.097294057
149:16  0.050522501
150:16 -0.041759476
151:16 -0.072763501
152:16 -0.317832843
153:16  0.114315240
154:17  0.086632745
155:17  0.099802640
156:17 -0.742330870
157:17  0.232990634
158:17 -0.013245408
159:17 -0.458051441
160:17  0.023670396
161:18  0.220777174
162:18  0.031607442
163:18 -0.044274052
164:19  0.163987540
```

```
165:20  0.244541493
166:20 -0.044615045
167:20  0.069234334
168:20 -0.000447702
169:20  0.095869690
170:20 -0.180875168
171:20  0.069234334
172:21  0.179569731
173:22  0.137941329
174:22 -0.176076642
175:23  0.050173494
176:24  0.288426131
177:24 -0.035329477
178:24 -0.189436585
179:25 -0.148315353
180:25 -0.154334267
181:25 -0.550995029
182:25 -0.024613086
183:25 -0.379335074
184:25  0.289335690
185:25  0.295998069

$country
    (Intercept)
1  -0.003018853
2  -0.029180642
3   0.005041075
4   0.046732147
5  -0.089924732
6   0.065226856
7  -0.009632653
8  -0.018993616
9   0.082720178
10 -0.005571528
11 -0.008233818
12  0.011266513
13 -0.004291477
14 -0.079616868
15 -0.043702196
16 -0.006329281
17 -0.040267015
18  0.010875601
19  0.008569786
20  0.013218433
21  0.009384092
22 -0.001992904
23  0.002622005
24  0.003326797
25 -0.035131428

with conditional variances for "site:country" "country"
```

```r
predFun <- function(fit0) {
    predict(fit0)
```

```
}
bb <- bootMer(fit0, nsim = 200, FUN = predFun, seed = 101)

# https://stats.stackexchange.com/questions/147836/prediction-interval-for-lmer-mixed-effects-model-in-
c(attr(ranef(fit0, condVar = TRUE)[[1]], "postVar"))  # site variances
```

```
  [1] 0.22916761 0.17457031 0.16731944 0.17560474 0.18538175
  [6] 0.20823773 0.24903904 0.23752324 0.16493834 0.19942799
 [11] 0.19229567 0.12251010 0.23508401 0.18187609 0.21906933
 [16] 0.18187609 0.11933817 0.09327248 0.19942799 0.20823773
 [21] 0.19276272 0.17001378 0.18726908 0.24048261 0.19912527
 [26] 0.20554568 0.19113444 0.20137049 0.21117833 0.17434653
 [31] 0.17497344 0.13703589 0.20225073 0.15574951 0.18621404
 [36] 0.18768208 0.22798098 0.12688350 0.22477078 0.14164812
 [41] 0.17243539 0.16203747 0.13124648 0.16519516 0.13433509
 [46] 0.19789504 0.18610555 0.16847625 0.18109658 0.22014158
 [51] 0.18109658 0.18621404 0.23143641 0.17083526 0.18621404
 [56] 0.18366872 0.22264741 0.17276979 0.16131238 0.15722452
 [61] 0.17031768 0.18798697 0.14051384 0.21047343 0.23005777
 [66] 0.18912842 0.22264741 0.15897739 0.14153428 0.17856870
 [71] 0.18193630 0.20517760 0.16400760 0.24125602 0.17578310
 [76] 0.19191602 0.15473955 0.21084084 0.23863779 0.14702566
 [81] 0.19480394 0.15781238 0.17967325 0.19977568 0.14110976
 [86] 0.24771152 0.17641242 0.15021947 0.19844919 0.17641242
 [91] 0.22567841 0.08829245 0.19802839 0.14326935 0.23257646
 [96] 0.17842090 0.20328994 0.22567841 0.19355673 0.17036183
[101] 0.16446407 0.19023950 0.18363382 0.22178545 0.17877632
[106] 0.15341032 0.13687302 0.18791438 0.13528385 0.18271062
[111] 0.22031951 0.23960818 0.10459948 0.14351492 0.19189560
[116] 0.24875168 0.23463943 0.19189560 0.12632817 0.18648932
[121] 0.18135771 0.20242579 0.22812574 0.15417850 0.16223303
[126] 0.12612340 0.19214710 0.18648932 0.19654927 0.21812766
[131] 0.17846557 0.17767134 0.19914254 0.22861310 0.15672471
[136] 0.22861310 0.17767134 0.22508836 0.20313019 0.13267137
[141] 0.13957071 0.23997893 0.19187728 0.17353495 0.15881819
[146] 0.21753944 0.19392581 0.24035859 0.24917144 0.22150748
[151] 0.19285982 0.10823922 0.13653666 0.17561597 0.24001684
[156] 0.19195128 0.18525813 0.16341636 0.16455727 0.14317200
[161] 0.10911788 0.15898615 0.22180107 0.16763522 0.16013200
[166] 0.22182887 0.17686250 0.18435336 0.24055916 0.17887326
[171] 0.17686250 0.22599018 0.17020878 0.17864354 0.24922503
[176] 0.20915790 0.18845104 0.15917690 0.14052488 0.20330634
[181] 0.19509215 0.18756431 0.13034668 0.15671691 0.20828096
```

```
c(attr(ranef(fit0, condVar = TRUE)[[2]], "postVar"))  # country variances
```

```
 [1] 0.012999595 0.010429581 0.013498750 0.012393032
 [5] 0.009573457 0.011671042 0.013276161 0.013228525
 [9] 0.011123841 0.012369155 0.012109529 0.011765868
[13] 0.012833154 0.010319275 0.011622184 0.011847956
[17] 0.012175361 0.012790224 0.013299806 0.012348947
[21] 0.013459172 0.013094500 0.013522626 0.012963344
[25] 0.012080273
```

Use sandwich approach in reference , although I think it can only handle one cluster

```
require(rms)
dd <- datadist(d)  #  Run for all potential vars.
options(datadist = "dd")

d$tx <- rnorm(sum(persons), 0, 1)  # need to do this otherwise intercept model has no covar matrix
o <- try(lrm(y ~ 1 + tx, x = TRUE, y = TRUE, d))
(v <- robcov(fit = o, cluster = d[, c(2)]))  # can I cluster on multiple variances - no?
```

```
Logistic Regression Model

 lrm(formula = y ~ 1 + tx, data = d, x = TRUE, y = TRUE)

                          Model Likelihood     Discrimination    Rank Discrim.
                             Ratio Test           Indexes           Indexes
 Obs              2090    LR chi2       0.03    R2        0.000    C       0.505
  0                441    d.f.             1    g         0.011    Dxy     0.010
  1               1649    Pr(> chi2) 0.8612     gr        1.011    gamma   0.010
 Cluster on  d[, c(2)]                          gp        0.002    tau-a   0.003
 Clusters           25                          Brier     0.166
 max |deriv|     2e-14

           Coef   S.E.   Wald Z Pr(>|Z|)
 Intercept 1.3189 0.0730 18.07  <0.0001
 tx        0.0095 0.0554  0.17  0.8637
```

```
summary(v)
```

```
             Effects              Response : y

 Factor      Low High Diff. Effect S.E. Lower 0.95
 tx          1   1    0     0       0   0
  Odds Ratio 1   1    0     1      NA   1
 Upper 0.95
 0
 1
```

Analysis ignoring clustering

```
binom::binom.confint(sum(d$y == 1), length(d$y))
```

```
          method    x    n      mean     lower     upper
1  agresti-coull 1649 2090 0.7889952 0.7709722 0.8059578
2     asymptotic 1649 2090 0.7889952 0.7715025 0.8064880
3          bayes 1649 2090 0.7888570 0.7712979 0.8062611
4        cloglog 1649 2090 0.7889952 0.7708696 0.8058727
5          exact 1649 2090 0.7889952 0.7708638 0.8063123
6          logit 1649 2090 0.7889952 0.7709717 0.8059573
7         probit 1649 2090 0.7889952 0.7710815 0.8060594
8        profile 1649 2090 0.7889952 0.7711547 0.8061277
9            lrt 1649 2090 0.7889952 0.7711318 0.8061254
10     prop.test 1649 2090 0.7889952 0.7707342 0.8061813
```

```
11        wilson 1649 2090 0.7889952 0.7709803 0.8059498
```

Bayesian

```r
require(brms)
require(rstan)

rstan_options(auto_write = TRUE)
# options(mc.cores = parallel::detectCores())

fit = brm(formula = y ~ (1 | country/site), family = bernoulli,
    data = d, seed = 123, )
```

```
SAMPLING FOR MODEL '5eccc82172b36984db900788f26aa760' NOW (CHAIN 1).
Chain 1:
Chain 1: Gradient evaluation took 0.001 seconds
Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 10 seconds.
Chain 1: Adjust your expectations accordingly!
Chain 1:
Chain 1:
Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
Chain 1:
Chain 1:  Elapsed Time: 13.542 seconds (Warm-up)
Chain 1:                16.416 seconds (Sampling)
Chain 1:                29.958 seconds (Total)
Chain 1:

SAMPLING FOR MODEL '5eccc82172b36984db900788f26aa760' NOW (CHAIN 2).
Chain 2:
Chain 2: Gradient evaluation took 0 seconds
Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
Chain 2: Adjust your expectations accordingly!
Chain 2:
Chain 2:
Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
```

```
Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
Chain 2:
Chain 2:  Elapsed Time: 12.787 seconds (Warm-up)
Chain 2:                13.757 seconds (Sampling)
Chain 2:                26.544 seconds (Total)
Chain 2:

SAMPLING FOR MODEL '5eccc82172b36984db900788f26aa760' NOW (CHAIN 3).
Chain 3:
Chain 3: Gradient evaluation took 0 seconds
Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
Chain 3: Adjust your expectations accordingly!
Chain 3:
Chain 3:
Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
Chain 3:
Chain 3:  Elapsed Time: 11.545 seconds (Warm-up)
Chain 3:                12.504 seconds (Sampling)
Chain 3:                24.049 seconds (Total)
Chain 3:

SAMPLING FOR MODEL '5eccc82172b36984db900788f26aa760' NOW (CHAIN 4).
Chain 4:
Chain 4: Gradient evaluation took 0 seconds
Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
Chain 4: Adjust your expectations accordingly!
Chain 4:
Chain 4:
Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
```

```
Chain 4:
Chain 4:  Elapsed Time: 12.318 seconds (Warm-up)
Chain 4:                7.491 seconds (Sampling)
Chain 4:                19.809 seconds (Total)
Chain 4:
```

```r
print(fit)
```

```
 Family: bernoulli
  Links: mu = logit
Formula: y ~ (1 | country/site)
   Data: d (Number of observations: 2090)
Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
         total post-warmup samples = 4000

Group-Level Effects:
~country (Number of levels: 25)
              Estimate Est.Error l-95% CI u-95% CI
sd(Intercept)     0.16      0.10     0.01     0.39
              Eff.Sample Rhat
sd(Intercept)        987 1.00

~country:site (Number of levels: 185)
              Estimate Est.Error l-95% CI u-95% CI
sd(Intercept)     0.53      0.10     0.34     0.72
              Eff.Sample Rhat
sd(Intercept)        965 1.01

Population-Level Effects:
          Estimate Est.Error l-95% CI u-95% CI Eff.Sample
Intercept     1.39      0.09     1.22     1.56       3267
          Rhat
Intercept 1.00

Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
is a crude measure of effective sample size, and Rhat is the potential
scale reduction factor on split chains (at convergence, Rhat = 1).
```

```r
f <- fit  # selected model, default priors used

# save the workspace save.image(file= 'brms bernouli
# model.RData' )
```

Load Bayesian analysis

```r
# load(file='brms bernouli model.RData' ) #site sd=2.5,
# country=0.5 library(brms)
```

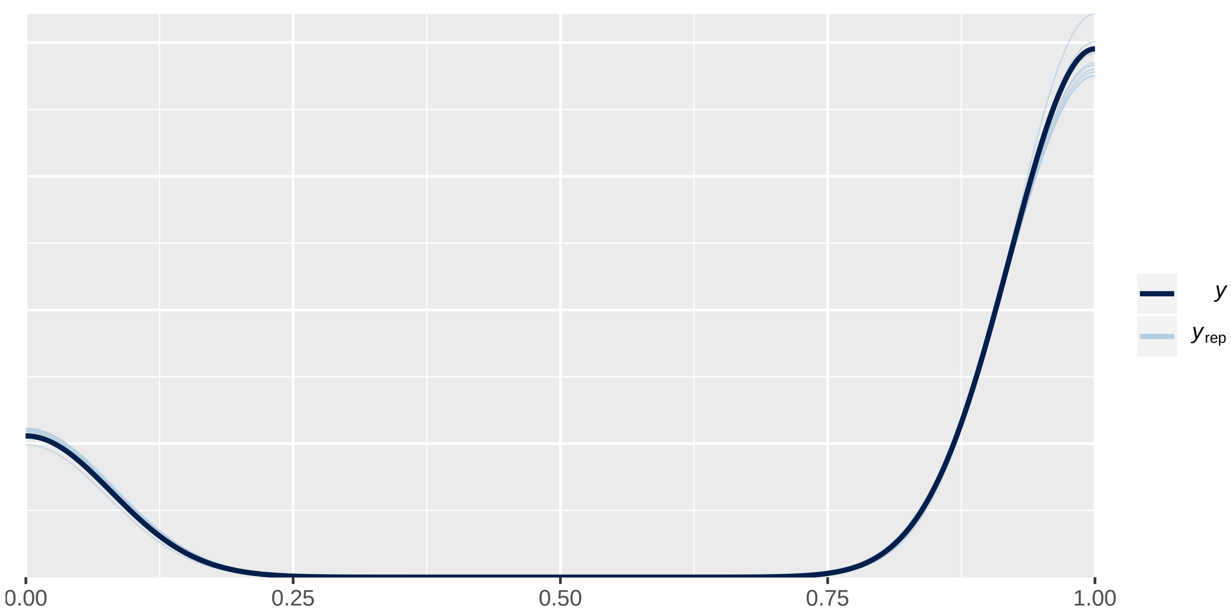
Check Model

```r
print(f)
```

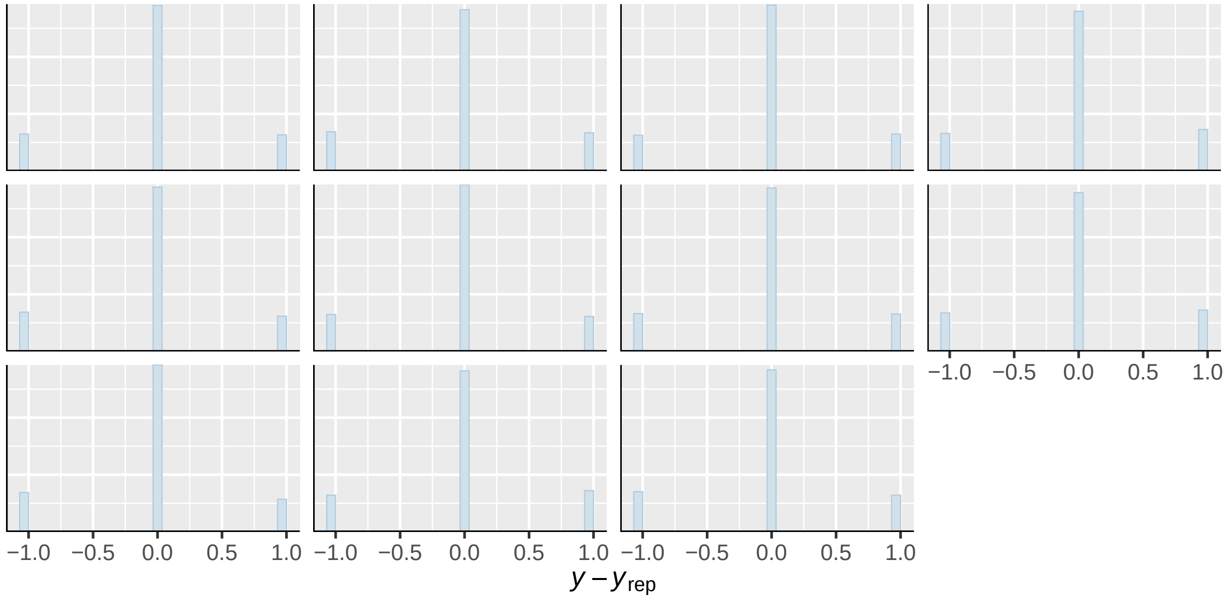
```
 Family: bernoulli
  Links: mu = logit
```

```
Formula: y ~ (1 | country/site)
   Data: d (Number of observations: 2090)
Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
         total post-warmup samples = 4000

Group-Level Effects:
~country (Number of levels: 25)
              Estimate Est.Error l-95% CI u-95% CI
sd(Intercept)     0.16      0.10     0.01     0.39
              Eff.Sample Rhat
sd(Intercept)        987 1.00

~country:site (Number of levels: 185)
              Estimate Est.Error l-95% CI u-95% CI
sd(Intercept)     0.53      0.10     0.34     0.72
              Eff.Sample Rhat
sd(Intercept)        965 1.01

Population-Level Effects:
          Estimate Est.Error l-95% CI u-95% CI Eff.Sample
Intercept     1.39      0.09     1.22     1.56       3267
          Rhat
Intercept 1.00

Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
is a crude measure of effective sample size, and Rhat is the potential
scale reduction factor on split chains (at convergence, Rhat = 1).
```

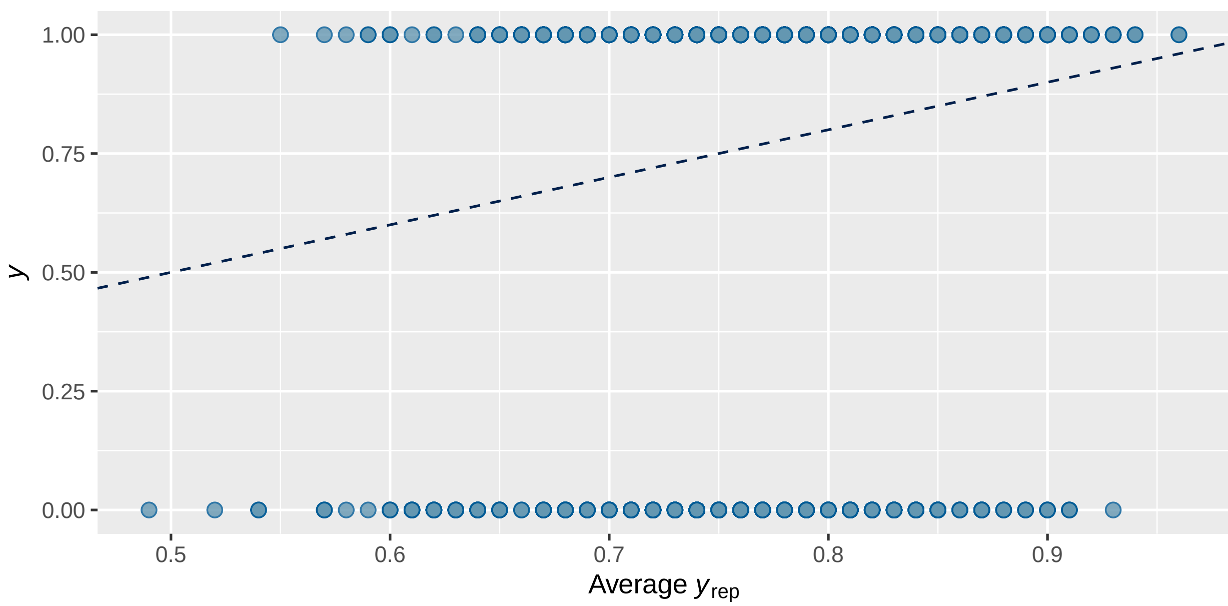
```r
pp_check(f)  # shows dens_overlay plot by default
```



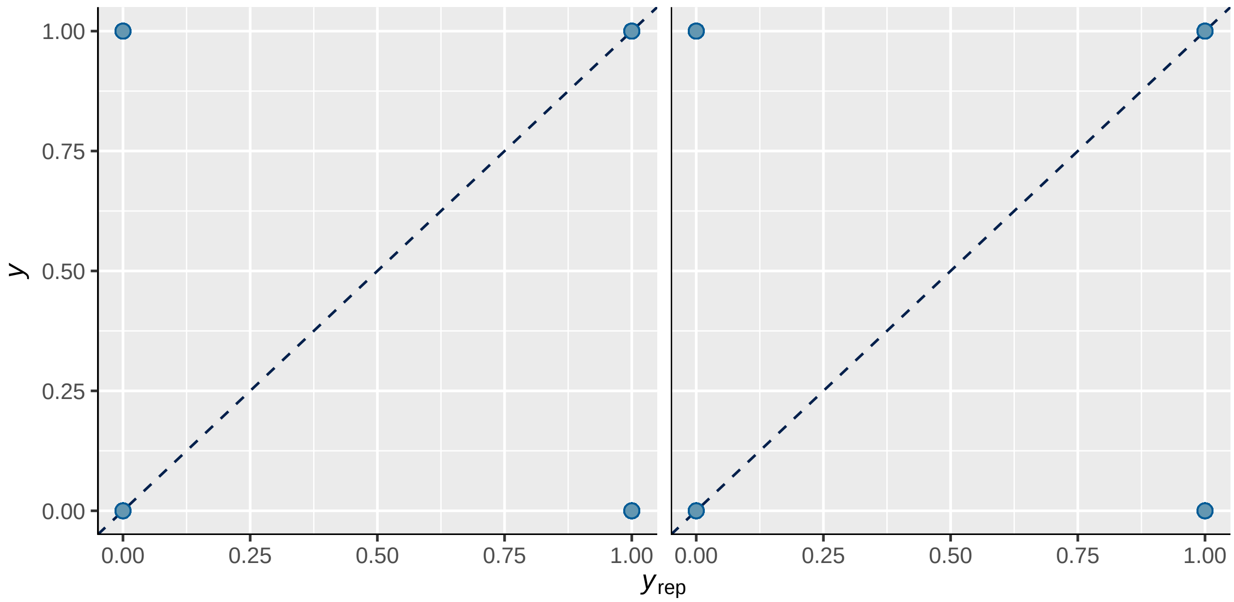

```r
pp_check(f, type = "error_hist", nsamples = 11)  # 11 draws
```



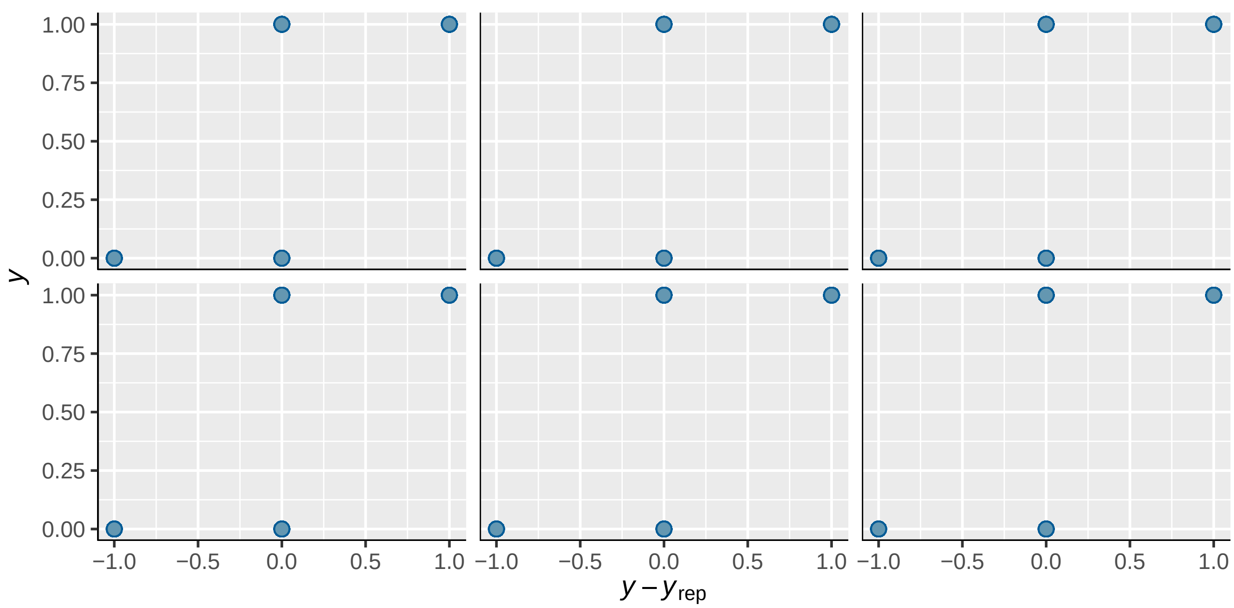

```r
pp_check(f, type = "scatter_avg", nsamples = 100)  # mean on x axis y observed
```



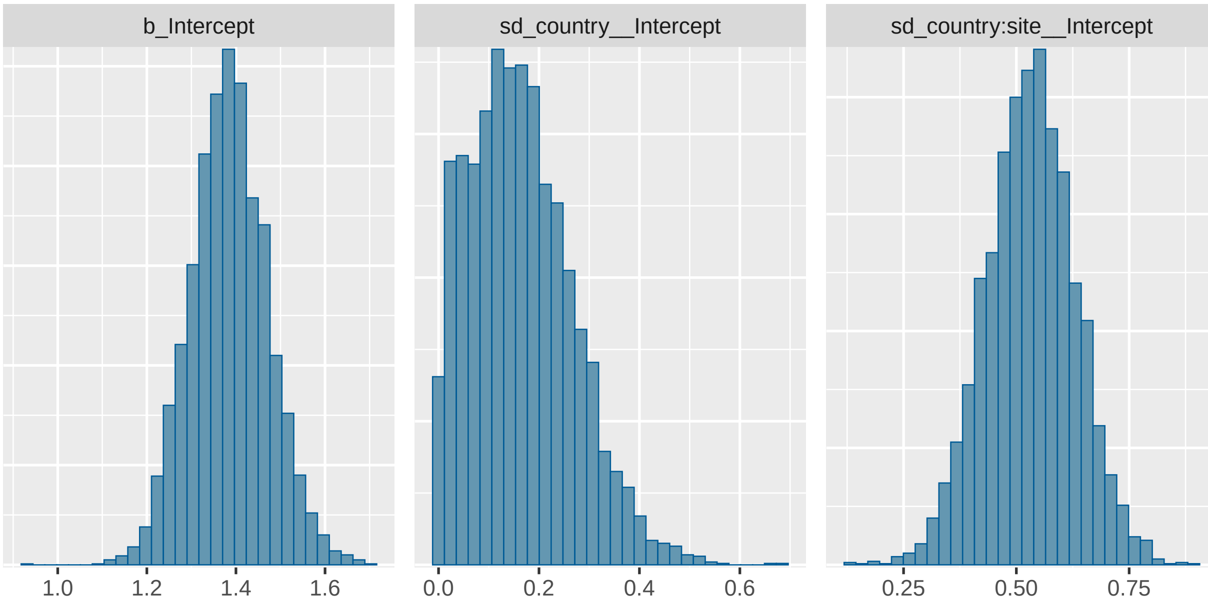

```r
# pp_check(f, type = 'stat_2d') ## took a long time so
# cancelled this pp_check(f ,x='vas', type = 'intervals') ##
# took a long time so cancelled this
pp_check(f, type = "scatter", nsamples = 2)
```




```
# pp_check(f,x='avisit',type = 'error_scatter_avg_vs_x',
# nsamples = 10) #throwing error pp_check(f,x='vas', type =
# 'ribbon', nsamples = 20) #throwing error
pp_check(f, type = "error_scatter", nsamples = 6)
```




```
stanplot(f, type = "hist")
```



Predictions

```r
print(f)
```

```
 Family: bernoulli
  Links: mu = logit
Formula: y ~ (1 | country/site)
   Data: d (Number of observations: 2090)
Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
         total post-warmup samples = 4000

Group-Level Effects:
~country (Number of levels: 25)
              Estimate Est.Error l-95% CI u-95% CI
sd(Intercept)     0.16      0.10     0.01     0.39
              Eff.Sample Rhat
sd(Intercept)        987 1.00

~country:site (Number of levels: 185)
              Estimate Est.Error l-95% CI u-95% CI
sd(Intercept)     0.53      0.10     0.34     0.72
              Eff.Sample Rhat
sd(Intercept)        965 1.01

Population-Level Effects:
          Estimate Est.Error l-95% CI u-95% CI Eff.Sample
Intercept     1.39      0.09     1.22     1.56       3267
          Rhat
Intercept 1.00

Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
is a crude measure of effective sample size, and Rhat is the potential
scale reduction factor on split chains (at convergence, Rhat = 1).
```

```r
f1 <- fitted(f, re_formula = NULL)  # include all random effects
head(f1)  #site level
```

```
      Estimate  Est.Error      Q2.5     Q97.5
[1,] 0.7715160 0.09005742 0.5605214 0.9104280
[2,] 0.7715160 0.09005742 0.5605214 0.9104280
[3,] 0.7715160 0.09005742 0.5605214 0.9104280
[4,] 0.7683298 0.07676172 0.5973683 0.8932983
[5,] 0.7683298 0.07676172 0.5973683 0.8932983
[6,] 0.7683298 0.07676172 0.5973683 0.8932983
```

```r
f2 <- fitted(f, re_formula = NA)  # no random effects
head(f2)  # intercept only
```

```
      Estimate  Est.Error      Q2.5     Q97.5
[1,] 0.7994577 0.01387825 0.7723696 0.8263722
[2,] 0.7994577 0.01387825 0.7723696 0.8263722
[3,] 0.7994577 0.01387825 0.7723696 0.8263722
[4,] 0.7994577 0.01387825 0.7723696 0.8263722
[5,] 0.7994577 0.01387825 0.7723696 0.8263722
[6,] 0.7994577 0.01387825 0.7723696 0.8263722
```

```r
f3 <- predict(f, re_formula = NULL)  # include all random effects
head(f3)  #individual
```

```
     Estimate Est.Error Q2.5 Q97.5
[1,]  0.77875 0.4151404    0     1
[2,]  0.77925 0.4148041    0     1
[3,]  0.76125 0.4263729    0     1
[4,]  0.76975 0.4210454    0     1
[5,]  0.76600 0.4234251    0     1
[6,]  0.77650 0.4166427    0     1
```

```r
f4 <- predict(f, re_formula = NA)  # no random effects
head(f4)  #individual
```

```
     Estimate Est.Error Q2.5 Q97.5
[1,]  0.80000 0.4000500    0     1
[2,]  0.79650 0.4026516    0     1
[3,]  0.79025 0.4071810    0     1
[4,]  0.79950 0.4004246    0     1
[5,]  0.79050 0.4070027    0     1
[6,]  0.79750 0.4019131    0     1
```

```r
f5 <- predict(f, re_formula = ~(1 | country))
head(f5)  #not sure
```

```
     Estimate Est.Error Q2.5 Q97.5
[1,]  0.79200 0.4059276    0     1
[2,]  0.79600 0.4030194    0     1
[3,]  0.79250 0.4055673    0     1
[4,]  0.79450 0.4041170    0     1
[5,]  0.80475 0.3964426    0     1
[6,]  0.80100 0.3992980    0     1
```

```r
f6 <- predict(f, re_formula = ~(1 | site))
head(f6)  # individual level
```

```
     Estimate Est.Error Q2.5 Q97.5
[1,]  0.79400 0.4044810    0     1
[2,]  0.79975 0.4002374    0     1
[3,]  0.79500 0.4037521    0     1
[4,]  0.80950 0.3927446    0     1
[5,]  0.80225 0.3983524    0     1
[6,]  0.80075 0.3994863    0     1
```

```
f8 <- fitted(f, re_formula = ~(1 | country))
head(f8)  #country level
```

```
      Estimate  Est.Error      Q2.5     Q97.5
[1,] 0.7977003 0.03055361 0.7294381 0.8565288
[2,] 0.7977003 0.03055361 0.7294381 0.8565288
[3,] 0.7977003 0.03055361 0.7294381 0.8565288
[4,] 0.7977003 0.03055361 0.7294381 0.8565288
[5,] 0.7977003 0.03055361 0.7294381 0.8565288
[6,] 0.7977003 0.03055361 0.7294381 0.8565288
```

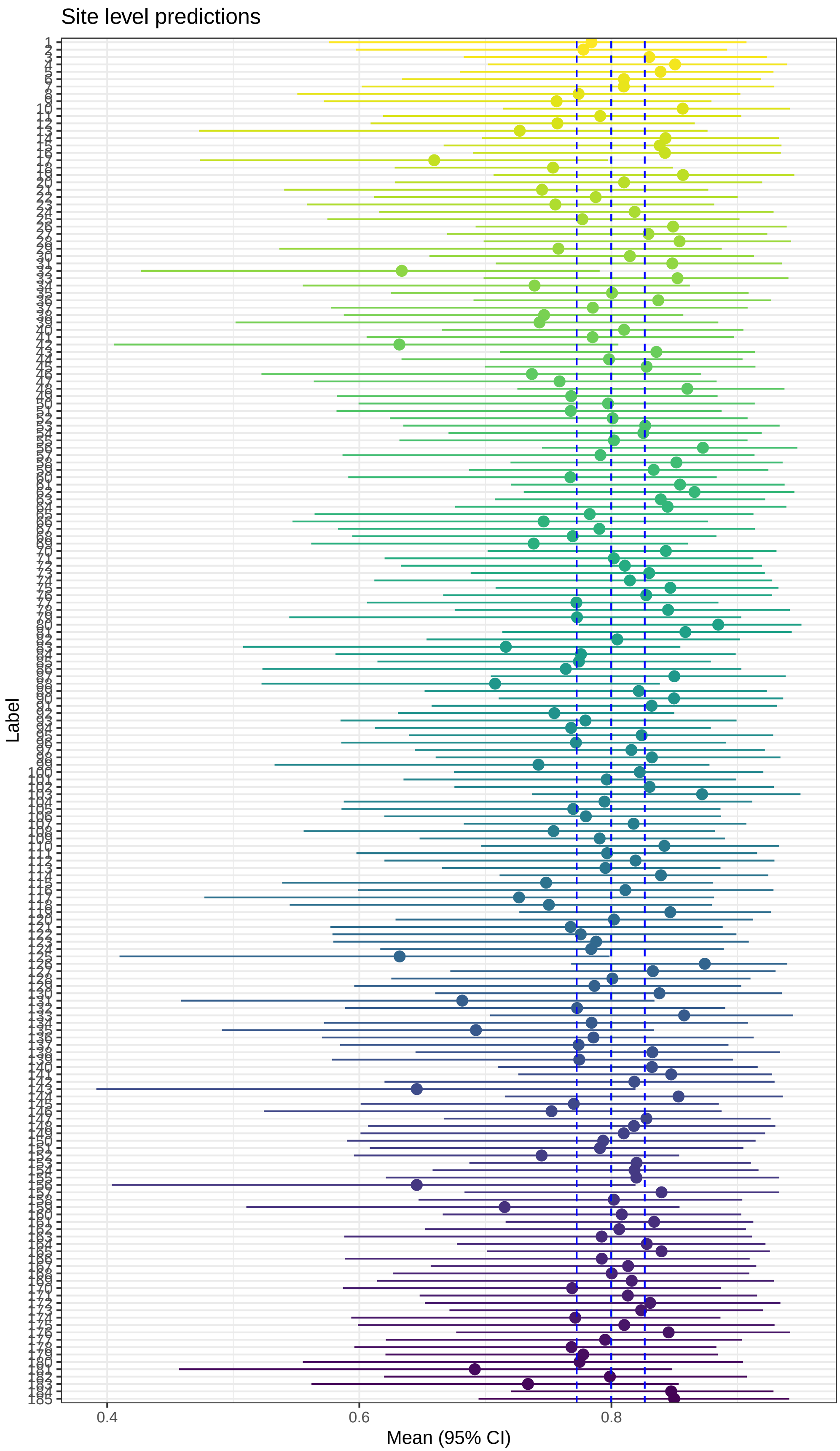
```
f9 <- fitted(f, re_formula = ~(1 | country/site))
head(f9)  #site level only same as f1
```

```
      Estimate  Est.Error      Q2.5     Q97.5
[1,] 0.7715160 0.09005742 0.5605214 0.9104280
[2,] 0.7715160 0.09005742 0.5605214 0.9104280
[3,] 0.7715160 0.09005742 0.5605214 0.9104280
[4,] 0.7683298 0.07676172 0.5973683 0.8932983
[5,] 0.7683298 0.07676172 0.5973683 0.8932983
[6,] 0.7683298 0.07676172 0.5973683 0.8932983
```

Plot Bayesian site level predictions

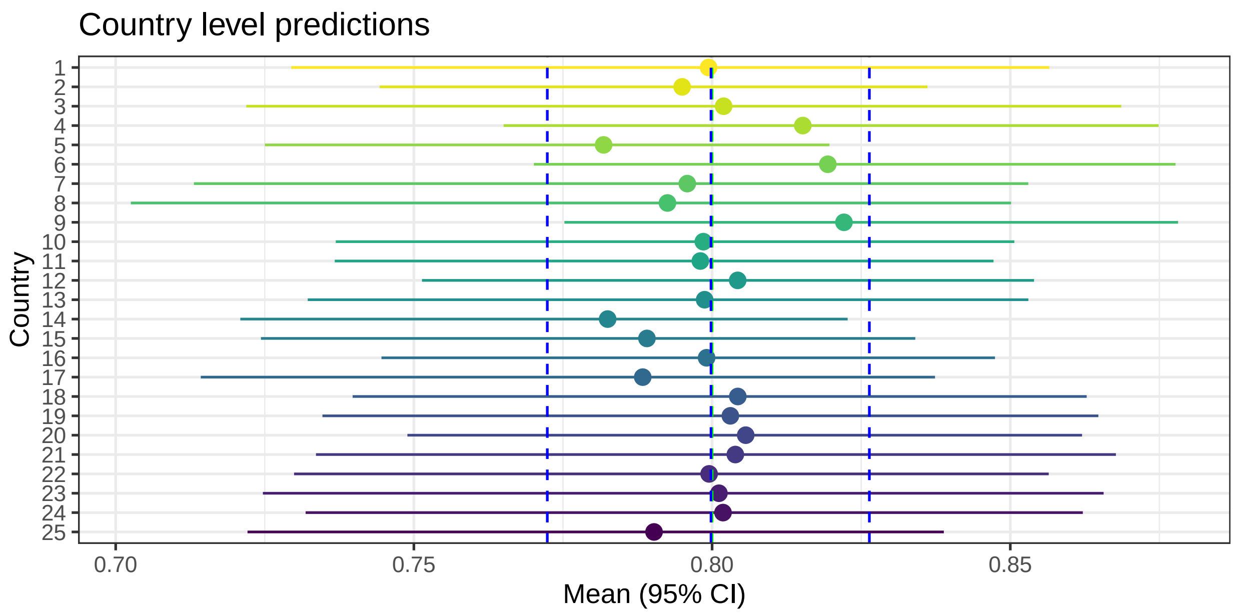
```
    pred <- f1 # f9
    df <- data.frame(cbind(pred, d))
    df$label <- df$site

 # start here use coefficients from model rather than predictions
    A <- function(x) 1/(1+exp(-x))

    e1 <- A(fixef(fit)[,'Estimate'])
    e2 <- A(fixef(fit)[,'Q2.5'])
    e3 <- A(fixef(fit)[,'Q97.5'])

    bc <- coef(fit, old=FALSE, summary=TRUE)$`country:site`
    df <- as.data.frame(bc)
    names(df) <- dimnames(bc)[[2]]
    df$sites <-     gsub(".*_", "", rownames(df))
    df$countries <- gsub("_.*", "", rownames(df))

    df <- data.frame(df[,c(2,5,6)], apply(df[,c(1,3,4)],2, A) )

    df <- df[,c(3,2,4,1,5,6)]

    df$label <- df$sites
    df$label <- as.numeric(as.character(df$label))
```

```
    df <- df[order(df$label),]
    # reverses the factor level ordering for labels after coord_flip()
   # df <- df[order(sites),]
    df$label <- factor(df$label, levels=rev(unique(df$label)), ordered = T)

    # df$label <- factor(   df$label, levels=unique(as.character(   df$label)) )

    fp <- NULL
    fp <- ggplot(data=df, aes(x=label, y=Estimate, ymin=Q2.5, ymax=Q97.5, colour=label)) +
      geom_pointrange() +
         geom_hline(yintercept=mu, color="green",lty=2) +  # add a dotted line

      geom_hline(yintercept=e1,  color="blue", linetype="dashed") +  # estimate
      geom_hline(yintercept=e2,   color="blue", linetype="dashed") +
      geom_hline(yintercept=e3,    color="blue", linetype="dashed") +
      coord_flip() +  # flip coordinates (puts labels on y axis)
      xlab("Label") + ylab("Mean (95% CI)") +
      theme_bw() + # use a white background
      theme(legend.position="none") +
     ggtitle("Site level predictions")
    print(fp)
```


Site level predictions
Label
Mean (95% CI)
0.4
0.6
0.8

Plot Bayesian country level predictions

```
#these are predictions
pred <- f8
df <- data.frame(cbind(pred, d))
df$label <- df$country

# start here use coefficients from model rather than predictions
bc <- coef(fit, old=FALSE, summary=TRUE)$country
df <- as.data.frame(bc)
names(df) <- dimnames(bc)[[2]]
df$label <- unique(d$country)


df <- data.frame(df[,c(2,5)], apply(df[,c(1,3,4)],2, A) )
df <- df[,c(2,3,1,4,5)]

# reverses the factor level ordering for labels after coord_flip()
df$label <- factor(df$label, levels=rev(unique(df$label)), ordered = T)

fp <- NULL
fp <- ggplot(data=df, aes(x=label, y=Estimate,
                          ymin=df[,4], ymax=df[,5], colour=label)) +
  geom_pointrange() +
  geom_hline(yintercept=mu, color="green",lty=2) +  # add a dotted line

  geom_hline(yintercept=e1, color="blue", linetype="dashed") +  # estimate
  geom_hline(yintercept=e2,    color="blue", linetype="dashed") +
  geom_hline(yintercept=e3,    color="blue", linetype="dashed") +

  coord_flip() +  # flip coordinates (puts labels on y axis)
  xlab("Country") +
  ylab("Mean (95% CI)") +
  theme_bw() + # use a white background
  theme(legend.position="none") +
  ggtitle("Country level predictions")
print(fp)
```




```r
# fitted_values <- fitted(fit)
# head(fitted_values)
# plot fitted means against actual response
# dat <- as.data.frame(cbind(Y = standata(fit)$Y, fitted_values))
# ggplot(dat) + geom_point(aes(x = Estimate, y = Y))
```

Exploring

```r
newdata <- data.frame(country = 1, site = 1, person = 1)
predict(fit, newdata = newdata)
```

```
     Estimate Est.Error Q2.5 Q97.5
[1,]    0.772 0.4195951    0     1
```

```r
f7 <- predict(f, re_formula = NULL, summary = F)  # include all random effects
f7[1:10, 1:10]  #columns are samples, rows predictions
```

```
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
 [1,]    1    1    1    1    0    0    1    1    1     1
 [2,]    1    1    0    1    1    1    0    1    1     1
 [3,]    0    1    1    1    1    0    1    1    0     0
 [4,]    1    1    1    1    1    1    1    0    1     1
 [5,]    1    1    1    1    1    1    1    1    1     0
 [6,]    1    1    1    1    1    1    1    1    1     0
 [7,]    1    1    1    1    0    1    0    0    1     1
 [8,]    0    0    1    1    1    1    1    1    1     0
 [9,]    1    0    1    0    1    0    1    1    1     1
[10,]    1    1    1    1    1    1    1    1    0     1
```

```r
x <- pp_check(f, nsamples = 1)
head(x$data)
```

```
# A tibble: 6 x 6
   y_id rep_id rep_label            is_y  is_y_label     value
  <int>  <int> <fct>                <lgl> <fct>          <dbl>
```

```
1     1      1 italic(y)[rep]  ( 1~ FALSE italic(y)[re~     1
2     2      1 italic(y)[rep]  ( 1~ FALSE italic(y)[re~     1
3     3      1 italic(y)[rep]  ( 1~ FALSE italic(y)[re~     1
4     4      1 italic(y)[rep]  ( 1~ FALSE italic(y)[re~     1
5     5      1 italic(y)[rep]  ( 1~ FALSE italic(y)[re~     1
6     6      1 italic(y)[rep]  ( 1~ FALSE italic(y)[re~     1
```

```
data1 <- make_standata(formula = y ~ (1 | country/site), family = bernoulli,
    data = d, )

data1$Y
```

```
   [1] 0 1 1 1 0 1 0 1 0 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1
  [28] 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1
  [55] 1 1 1 1 1 0 1 0 1 1 1 1 0 1 0 1 1 0 1 1 0 1 1 1 1 1 1
  [82] 1 1 1 1 1 1 1 1 0 1 0 1 0 1 0 1 1 1 1 1 1 1 1 0 1 1 0
 [109] 1 1 1 0 0 1 0 1 0 0 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1
 [136] 1 1 1 1 1 1 1 1 1 0 1 1 1 1 0 0 1 1 1 1 1 0 1 1 0 1 0
 [163] 0 0 0 1 0 0 0 1 0 1 1 0 1 1 0 0 1 0 0 1 1 1 0 0 1 1 1
 [190] 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 [217] 0 1 1 1 1 1 0 1 1 0 1 1 0 1 1 1 1 0 0 1 1 1 1 1 0 1 1
 [244] 0 0 1 0 1 1 1 1 0 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 [271] 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1
 [298] 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 0 0 1 1 1 0
 [325] 1 0 0 0 1 0 0 0 1 1 1 1 1 1 1 1 1 0 1 0 0 1 1 1 0 1 0
 [352] 1 1 1 1 1 1 0 1 0 1 1 1 1 1 1 0 1 1 1 1 1 1 0 1 1 1 0
 [379] 0 0 0 1 1 1 1 1 1 0 0 1 1 1 0 1 1 1 1 0 1 0 1 1 0 1 1
 [406] 1 1 1 1 1 1 1 1 1 0 1 1 0 0 1 1 1 0 0 1 0 1 1 1 1 1 0
 [433] 0 1 0 0 0 1 0 1 0 1 1 0 1 1 1 1 1 0 0 1 1 0 1 1 1 1 1
 [460] 1 1 1 1 1 1 1 1 1 1 1 0 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0
 [487] 1 1 1 1 0 1 0 1 1 1 1 1 1 1 1 1 1 1 0 1 1 0 1 0 1 0 1
 [514] 1 0 1 0 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1
 [541] 0 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 0 1 1 1 1
 [568] 1 1 1 1 1 1 1 1 0 1 1 0 1 1 0 1 1 1 1 1 0 1 1 1 1 1 1
 [595] 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 0 1 1 1 1 1 1 1
 [622] 1 1 1 1 1 1 1 1 0 1 1 0 1 1 1 1 1 1 1 1 0 1 1 0 1 1 1
 [649] 1 1 1 1 0 1 1 0 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1
 [676] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1
 [703] 1 1 0 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 0
 [730] 0 0 1 1 1 1 1 0 1 0 1 1 1 1 1 1 0 1 0 1 1 1 1 0 0 1 0
 [757] 1 1 0 0 1 1 0 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 0 1 1
 [784] 1 1 1 1 0 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1
 [811] 1 1 1 0 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 0 1
 [838] 1 1 1 1 0 1 1 0 1 1 1 1 0 1 0 1 1 1 1 1 1 1 1 1 1 1 1
 [865] 0 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1
 [892] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1
 [919] 1 1 0 1 1 0 1 1 1 0 0 0 1 0 1 1 0 0 1 1 1 1 1 1 1 0 1
 [946] 0 1 1 1 1 1 0 1 1 1 1 1 1 0 0 0 1 1 1 1 0 1 1 1 1 1 1
 [973] 1 1 1 0 1 1 0 1 1 0 1 0 1 0 1 1 0 1 1 1 0 1 1 1 1 1 1
[1000] 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 0 0 0 1 1
[1027] 0 0 1 1 1 1 1 1 0 0 1 0 1 1 1 1 1 1 1 1 1 0 1 1 1 1 0
[1054] 1 0 1 1 1 1 1 0 1 1 0 1 1 0 1 0 1 1 1 1 1 1 1 1 1 0 0
[1081] 1 1 1 0 1 1 1 0 0 1 1 1 1 1 0 1 1 1 0 1 1 1 1 1 1 1 1
[1108] 1 1 1 1 0 1 0 0 1 1 0 1 1 1 1 1 0 1 1 1 1 1 0 1 1 1 1
[1135] 1 1 1 1 0 0 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```
[1162] 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 0 1 0 0 1 1 1 0 1 0 1
[1189] 1 1 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 0 0
[1216] 1 1 1 0 1 1 1 0 0 0 1 1 1 1 0 1 0 1 1 1 1 1 1 0 1 1 1
[1243] 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1
[1270] 1 0 0 1 1 1 0 1 1 1 0 1 0 0 1 1 1 1 0 1 1 1 1 1 1 1 1
[1297] 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1
[1324] 1 1 0 1 1 0 0 0 1 1 1 0 0 0 1 1 0 1 0 1 1 1 1 1 1 1 1
[1351] 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 0 0 1 1 1 0 1 0 1 1 1
[1378] 1 1 0 1 1 1 0 1 1 1 1 1 0 0 0 1 1 0 1 1 1 1 0 1 1 1 1
[1405] 0 1 1 0 0 0 1 1 1 1 1 0 0 1 0 1 0 0 0 0 1 1 1 1 0 1 1
[1432] 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
[1459] 1 0 1 1 1 1 1 1 1 0 1 1 0 1 1 0 1 1 1 1 0 1 1 1 1 0 1
[1486] 1 1 1 1 1 1 0 1 0 0 1 0 0 0 1 1 0 1 1 1 1 1 0 1 1 1 1
[1513] 1 1 1 1 1 0 1 1 0 1 0 0 1 1 0 0 1 1 0 1 1 1 0 1 1 1 0
[1540] 1 1 0 1 1 0 1 1 1 1 0 1 0 1 1 1 1 1 1 1 1 1 1 1 0 1 1
[1567] 0 1 1 0 1 1 1 1 1 0 1 1 1 1 1 1 1 1 0 1 1 1 1 1 0 1 1
[1594] 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 0 1 0 0 0 0 1 1 1 1 1 1
[1621] 1 1 1 1 1 0 1 1 1 0 1 1 1 1 1 1 0 1 1 0 1 0 1 1 0 1 0
[1648] 1 1 1 0 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 0 0 1 1 1
[1675] 1 1 0 1 0 1 0 1 1 1 1 0 1 1 1 1 0 1 0 0 0 1 1 1 1 1 0
[1702] 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1 1 1 1 1 1 1 0 1 1 1 1
[1729] 1 1 1 1 1 0 1 1 0 1 1 0 1 0 0 0 0 1 1 1 1 1 1 0 1 1 1
[1756] 1 1 1 1 0 1 1 0 1 0 1 1 1 1 1 0 1 1 1 0 0 1 1 0 0 1 1
[1783] 1 1 1 0 1 1 1 1 0 1 0 1 1 1 1 1 1 0 1 0 1 1 1 1 1 0 1
[1810] 0 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 0 1 1 1 1
[1837] 1 1 1 1 1 1 1 1 1 1 0 1 1 0 1 1 1 1 1 1 1 0 1 1 0 1 1
[1864] 0 1 0 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 1 1
[1891] 1 1 1 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1
[1918] 1 0 1 0 1 1 1 1 1 0 0 1 1 0 1 1 1 1 1 1 1 0 1 1 0 1 1
[1945] 1 1 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 0 1 0
[1972] 1 0 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 0 1 0 1 1 1 1 0 1 0
[1999] 1 1 0 1 1 1 0 1 1 1 1 1 1 0 1 1 1 1 0 1 0 1 1 1 0 1 1
[2026] 0 0 1 1 0 1 0 0 1 0 1 0 1 1 1 1 1 0 1 1 1 0 0 1 1 0 1
[2053] 1 1 0 1 1 0 1 0 1 0 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1
[2080] 0 1 1 1 1 1 1 1 1 1 1
```

```r
#########################################################################

samples1 <- posterior_samples(fit, "^b")
head(samples1)
```

```
  b_Intercept
1    1.416227
2    1.302200
3    1.319786
4    1.395953
5    1.281657
6    1.278301
```

```r
# extract posterior samples of group-level standard
# deviations
samples2 <- posterior_samples(fit, "^sd_")
head(samples2)
```

```
  sd_country__Intercept sd_country:site__Intercept
1            0.02658279                  0.6135912
```

```
2           0.02005495                    0.3237502
3           0.02916379                    0.5304974
4           0.10857203                    0.4951511
5           0.13592008                    0.4006490
6           0.13360791                    0.5442208
```

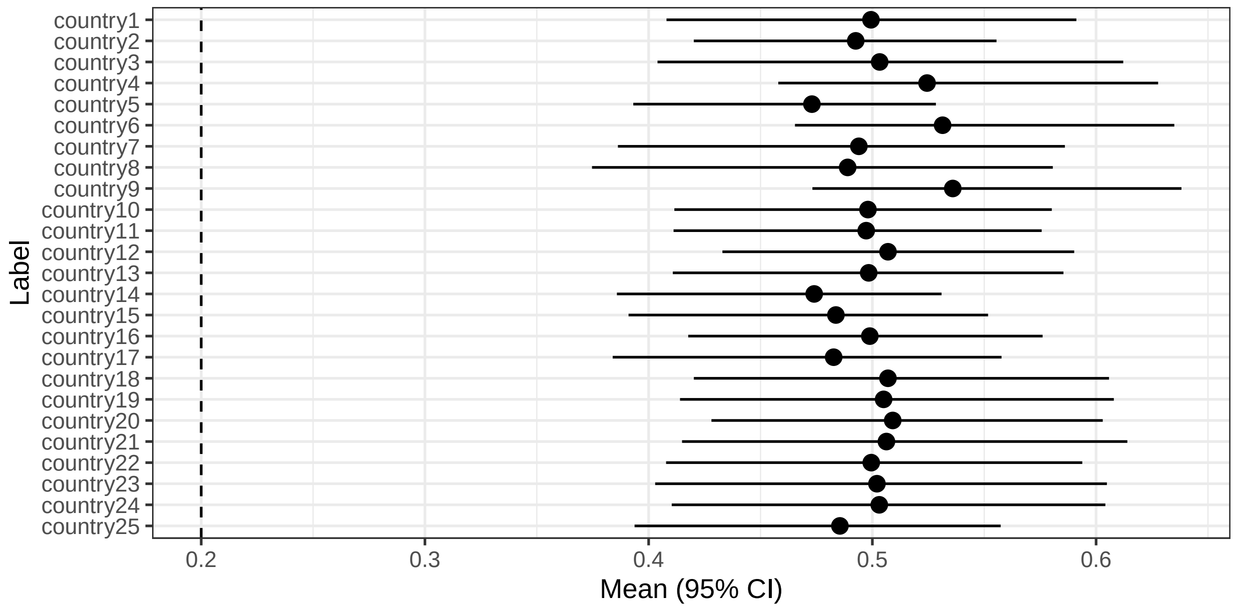
```
samples3 <- posterior_samples(fit, "^r_country")
head(samples3)[, 1:10]  # show forst 10 columns
```

```
  r_country[1,Intercept] r_country[2,Intercept]
1             0.04059608          -0.0018588509
2            -0.02694933           0.0004296924
3             0.01187981          -0.0014591342
4            -0.03449265          -0.1569559706
5             0.15947955           0.1161465576
6             0.09480129           0.0851873068
  r_country[3,Intercept] r_country[4,Intercept]
1          -0.0055018795           -0.009307143
2          -0.0008618772           -0.004757685
3          -0.0052542242            0.042872600
4          -0.1537275547           -0.206444167
5           0.2497969981            0.423036177
6           0.1374505506            0.136757827
  r_country[5,Intercept] r_country[6,Intercept]
1           -0.002700216           0.0133818052
2            0.001387618          -0.0091522255
3            0.001800244           0.0005819566
4           -0.149413207          -0.0232543323
5            0.093863217           0.2603559639
6            0.132755687           0.0619832284
  r_country[7,Intercept] r_country[8,Intercept]
1           -0.032616846             0.01204249
2            0.006723938             0.01850034
3           -0.002106794            -0.01417744
4           -0.153157739            -0.10972267
5            0.112950635             0.07058122
6           -0.105428485            -0.19266037
  r_country[9,Intercept] r_country[10,Intercept]
1           0.0038423018            0.0004800882
2          -0.0009913202           -0.0020800925
3           0.0102982595            0.0237693839
4           0.2243822261            0.1069214159
5          -0.0152885790           -0.1020087360
6           0.2407526882           -0.1057132940
```

Plot country SD estimates

```
  # posterior samples
  mc.1 <- as.mcmc(f, pars = NA, exact_match = TRUE,
                  combine_chains = TRUE, inc_warmup = FALSE)

  # get specific estimates
  # names(mc.1[1,])
  mc_country <- mc.1[, grep("r_country[", names(mc.1[1,]), fixed=TRUE) ]
```

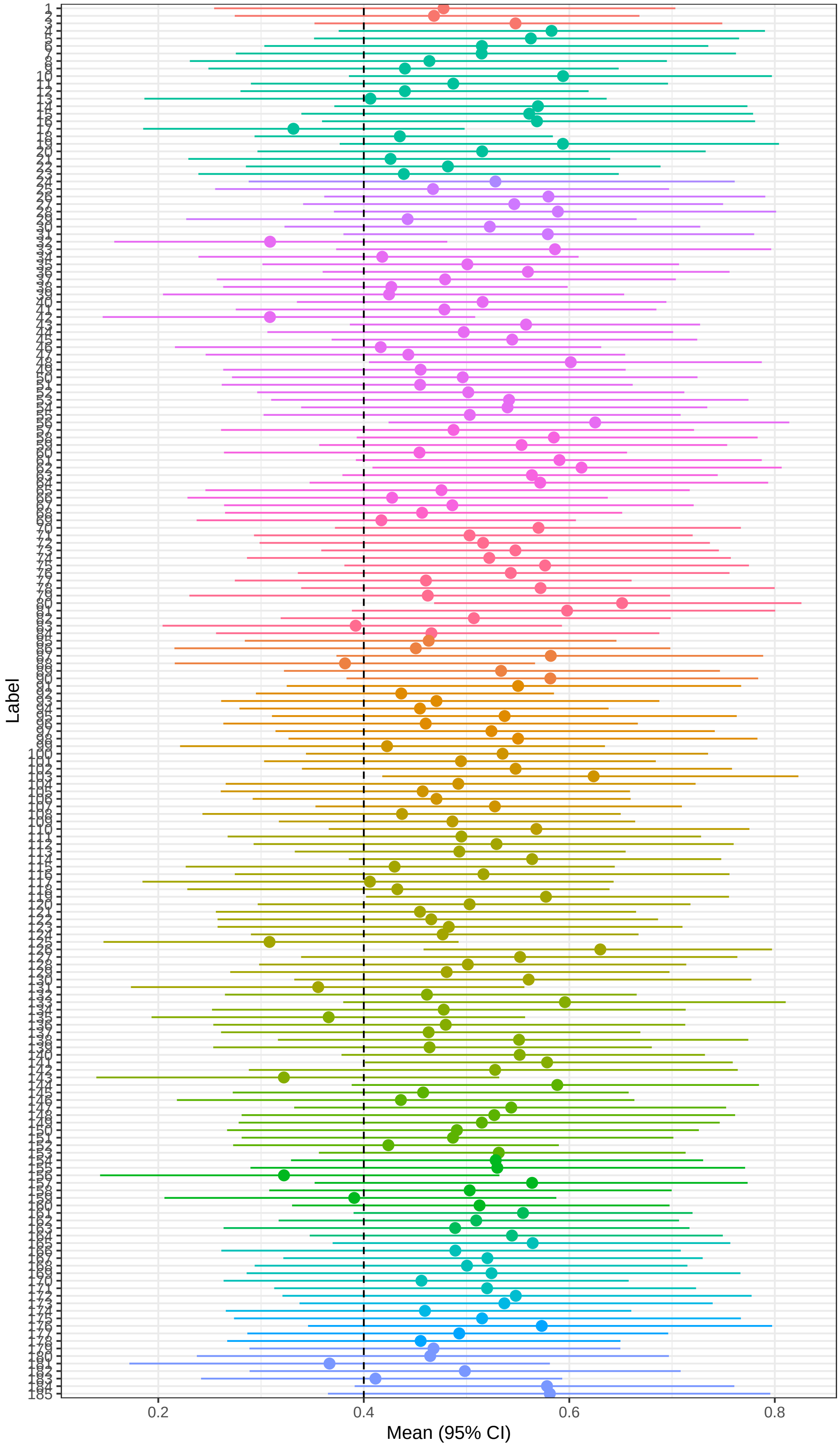
```
  # names(mc_country[1,])

  # convert to probabilities
  prob <- apply(mc_country,c(2),function(x) exp(x)/(1+exp(x)))
  # function to calculate summary stats
  statz <- function(x) {
                 t(cbind(c(mean(x), quantile(x, c(0.025,0.975))) ))
          }

  #here are the country specific estimates
  est <- apply(prob,2,statz)

 # forest plot see refernce
 label <- paste0("country", 1:25)
 mean  <-  est[1,]
 lower <-  est[2,]
 upper <-  est[3,]

 df <- data.frame(label, mean, lower, upper)

 # reverses the factor level ordering for labels after coord_flip()
 df$label <- factor(df$label, levels=rev(df$label))


 library(ggplot2)

 fp <- NULL
 fp <- ggplot(data=df, aes(x=label, y=mean, ymin=lower, ymax=upper)) +
         geom_pointrange() +
         geom_hline(yintercept=sdcountry, lty=2) +  # add a dotted line at x=1 after flip
         coord_flip() +  # flip coordinates (puts labels on y axis)
         xlab("Label") + ylab("Mean (95% CI)") +
         theme_bw()  # use a white background
 print(fp)
```



Plot site SD estimates

```
  # posterior samples
  mc.1 <- as.mcmc(f, pars = NA, exact_match = TRUE,
                  combine_chains = TRUE, inc_warmup = FALSE)

  # get specific estimates
  # names(mc.1[1,])
  mc_country <- mc.1[, grep("r_country:", names(mc.1[1,]), fixed=TRUE) ]
  #names(mc_country[1,])

  # convert to probabilities
  prob <- apply(mc_country, c(2) ,function(x) exp(x)/(1+exp(x)))

  # function to calculate summary stats
  statz <- function(x) {
                  t(cbind(c(mean(x), quantile(x, c(0.025,0.975))) ))
          }

  #here are the specific estimates
  est <- apply(prob,2,statz)

 # forest plot see reference
 # label <- paste0("site", 1:dim(mc_country)[2])

 #get labbeling info

 x1 <- gsub("[^0-9\\_]", "",  names(mc_country[1,]))
 x2 <- gsub("\\_", "",  x1)

 x2a <- gsub("\\_", ".",  x1)
 co <- gsub(".*\\.(.*)\\..*", "\\1", x2a)
```

```
site.only <- sub('.*\\.', '', x2a)
label <-  as.numeric(site.only)
mean  <-  est[1,]
lower <-  est[2,]
upper <-  est[3,]


df <- data.frame(co, label, mean, lower, upper)
df <- df[order(label),]

# reverses the factor level ordering for labels after coord_flip()
df$label <- factor(df$label, levels=rev(df$label))

fp <- NULL
fp <- ggplot(data=df, aes(x=label, y=mean, ymin=lower, ymax=upper, colour=co)) +
        geom_pointrange() +
        geom_hline(yintercept=sdsite, lty=2) +  # add a dotted line at x=1 after flip
        coord_flip() +  # flip coordinates (puts labels on y axis)
        xlab("Label") + ylab("Mean (95% CI)") +
        theme_bw() + # use a white background +
  theme(legend.position="none")

print(fp)
```


Label
Mean (95% CI)
0.2
0.4
0.6
0.8

Comparing frequentist and Bayesian, what the difference between fitted and coef from model output?

```
# fixef(fit, old=FALSE) object$fit@sim country predictions
# https://github.com/paul-buerkner/brms/issues/82

##########################################
f8 <- fitted(f, re_formula = ~(1 | country))  # is this the correct way for country level?
ba <- unique(f8)[, 1]  # Bayesian predictions
ba[1:10]


fr <- as.vector(unlist(1/(1 + exp(-coef(fit0)$country))))  # freq blups
plot(fr, ba)
abline(0, 1)  # compare Bayesian and Frequentist

########################################## above differs to model coefficients?
bc <- as.vector(coef(fit, old = TRUE)$country)
bc <- 1/(1 + exp(-bc))
bc

plot(fr, bc)
abline(0, 1)  # compare Bayesian and Frequentist

##########################################
f <- as.vector(unlist(coef(fit0)$country))  #frequentist
b <- as.vector(coef(fit)$country[, 1, ])  #bayes
plot(f, b)
abline(0, 1)
cor(f, b)


f <- as.vector(unlist(coef(fit0)$site))  #frequentist
# b <- as.vector(coef(fit)$`country:site`[,1,]) bayes , needs
# ordering
bc <- coef(fit, old = FALSE, summary = TRUE)$`country:site`
df <- as.data.frame(bc)
names(df) <- dimnames(bc)[[2]]
df$sites <- gsub(".*_", "", rownames(df))
# df$countries <- gsub('_.*', '', rownames(df))
df$sites <- as.numeric(as.character(df$sites))
df <- df[order(df$sites), ]
b <- df[, 1]
plot(f, b)
abline(0, 1)
cor(f, b)
```

Crude estimates

```
foo <- d

library(dplyr)
foox <- foo %>% group_by(country, site) %>% summarise(N = length(y),
```

```
    ones = (mean(y)) * length(y), zero = (1 - mean(y)) * length(y),
    mean = mean(y), )

foox <- data.frame(foox)
names(foox) <- c("Country", "site", "N", "Yes", "No", "Mean")

foox <- cbind(foox, binconf(foox$Yes, foox$N))
foox
```

```
      Country site  N Yes No      Mean  PointEst
X           1    1  3   2  1 0.6666667 0.6666667
X.1         1    2 11   8  3 0.7272727 0.7272727
X.2         1    3 15  13  2 0.8666667 0.8666667
X.3         2    4 14  13  1 0.9285714 0.9285714
X.4         2    5 11  10  1 0.9090909 0.9090909
X.5         2    6  6   5  1 0.8333333 0.8333333
X.6         2    7  1   1  0 1.0000000 1.0000000
X.7         2    8  2   1  1 0.5000000 0.5000000
X.8         2    9 12   8  4 0.6666667 0.6666667
X.9         2   10  9   9  0 1.0000000 1.0000000
X.10        2   11  8   6  2 0.7500000 0.7500000
X.11        2   12 24  17  7 0.7083333 0.7083333
X.12        2   13  2   0  2 0.0000000 0.0000000
X.13        2   14 12  11  1 0.9166667 0.9166667
X.14        2   15  5   5  0 1.0000000 1.0000000
X.15        2   16 12  11  1 0.9166667 0.9166667
X.16        2   17 21  11 10 0.5238095 0.5238095
X.17        2   18 39  28 11 0.7179487 0.7179487
X.18        2   19  9   9  0 1.0000000 1.0000000
X.19        2   20  6   5  1 0.8333333 0.8333333
X.20        2   21  7   4  3 0.5714286 0.5714286
X.21        2   22 12   9  3 0.7500000 0.7500000
X.22        2   23  8   5  3 0.6250000 0.6250000
X.23        3   24  2   2  0 1.0000000 1.0000000
X.24        4   25  7   5  2 0.7142857 0.7142857
X.25        4   26  8   8  0 1.0000000 1.0000000
X.26        4   27 10   9  1 0.9000000 0.9000000
X.27        4   28  9   9  0 1.0000000 1.0000000
X.28        4   29  5   3  2 0.6000000 0.6000000
X.29        4   30 13  11  2 0.8461538 0.8461538
X.30        4   31 15  14  1 0.9333333 0.9333333
X.31        5   32 15   6  9 0.4000000 0.4000000
X.32        5   33  8   8  0 1.0000000 1.0000000
X.33        5   34 13   8  5 0.6153846 0.6153846
X.34        5   35  9   7  2 0.7777778 0.7777778
X.35        5   36 10   9  1 0.9000000 0.9000000
X.36        5   37  3   2  1 0.6666667 0.6666667
X.37        5   38 21  14  7 0.6666667 0.6666667
X.38        5   39  3   1  2 0.3333333 0.3333333
X.39        5   40 20  16  4 0.8000000 0.8000000
X.40        5   41 11   8  3 0.7272727 0.7272727
X.41        5   42 10   3  7 0.3000000 0.3000000
X.42        5   43 26  22  4 0.8461538 0.8461538
X.43        5   44 13  10  3 0.7692308 0.7692308
```

X.44	5	45	24	20	4	0.8333333	0.8333333
X.45	5	46	6	3	3	0.5000000	0.5000000
X.46	5	47	8	5	3	0.6250000	0.6250000
X.47	5	48	16	15	1	0.9375000	0.9375000
X.48	5	49	9	6	3	0.6666667	0.6666667
X.49	5	50	4	3	1	0.7500000	0.7500000
X.50	5	51	9	6	3	0.6666667	0.6666667
X.51	5	52	9	7	2	0.7777778	0.7777778
X.52	5	53	3	3	0	1.0000000	1.0000000
X.53	5	54	13	11	2	0.8461538	0.8461538
X.54	5	55	9	7	2	0.7777778	0.7777778
X.55	5	56	13	13	0	1.0000000	1.0000000
X.56	6	57	4	3	1	0.7500000	0.7500000
X.57	6	58	16	15	1	0.9375000	0.9375000
X.58	6	59	18	16	2	0.8888889	0.8888889
X.59	6	60	15	11	4	0.7333333	0.7333333
X.60	6	61	17	16	1	0.9411765	0.9411765
X.61	6	62	13	13	0	1.0000000	1.0000000
X.62	6	63	26	23	3	0.8846154	0.8846154
X.63	6	64	7	7	0	1.0000000	1.0000000
X.64	6	65	3	2	1	0.6666667	0.6666667
X.65	6	66	8	5	3	0.6250000	0.6250000
X.66	6	67	4	3	1	0.7500000	0.7500000
X.67	7	68	14	10	4	0.7142857	0.7142857
X.68	8	69	17	11	6	0.6470588	0.6470588
X.69	9	70	14	13	1	0.9285714	0.9285714
X.70	9	71	11	9	2	0.8181818	0.8181818
X.71	9	72	7	6	1	0.8571429	0.8571429
X.72	9	73	17	15	2	0.8823529	0.8823529
X.73	9	74	2	2	0	1.0000000	1.0000000
X.74	9	75	15	14	1	0.9333333	0.9333333
X.75	9	76	10	9	1	0.9000000	0.9000000
X.76	9	77	16	12	4	0.7500000	0.7500000
X.77	9	78	7	7	0	1.0000000	1.0000000
X.78	9	79	2	1	1	0.5000000	0.5000000
X.79	9	80	31	30	1	0.9677419	0.9677419
X.80	9	81	11	11	0	1.0000000	1.0000000
X.81	9	82	17	14	3	0.8235294	0.8235294
X.82	9	83	9	5	4	0.5555556	0.5555556
X.83	9	84	7	5	2	0.7142857	0.7142857
X.84	10	85	19	14	5	0.7368421	0.7368421
X.85	10	86	1	0	1	0.0000000	0.0000000
X.86	10	87	14	13	1	0.9285714	0.9285714
X.87	10	88	14	8	6	0.5714286	0.5714286
X.88	10	89	8	7	1	0.8750000	0.8750000
X.89	10	90	14	13	1	0.9285714	0.9285714
X.90	11	91	4	4	0	1.0000000	1.0000000
X.91	11	92	44	32	12	0.7272727	0.7272727
X.92	11	93	7	5	2	0.7142857	0.7142857
X.93	11	94	18	13	5	0.7222222	0.7222222
X.94	11	95	3	3	0	1.0000000	1.0000000
X.95	11	96	10	7	3	0.7000000	0.7000000
X.96	11	97	7	6	1	0.8571429	0.8571429
X.97	11	98	4	4	0	1.0000000	1.0000000

X.98	12	99	7	4	3	0.5714286	0.5714286
X.99	12	100	14	12	2	0.8571429	0.8571429
X.100	12	101	14	11	3	0.7857143	0.7857143
X.101	12	102	10	9	1	0.9000000	0.9000000
X.102	12	103	14	14	0	1.0000000	1.0000000
X.103	12	104	4	3	1	0.7500000	0.7500000
X.104	12	105	10	7	3	0.7000000	0.7000000
X.105	12	106	16	12	4	0.7500000	0.7500000
X.106	12	107	24	20	4	0.8333333	0.8333333
X.107	13	108	8	5	3	0.6250000	0.6250000
X.108	13	109	22	17	5	0.7727273	0.7727273
X.109	13	110	12	11	1	0.9166667	0.9166667
X.110	14	111	4	3	1	0.7500000	0.7500000
X.111	14	112	2	2	0	1.0000000	1.0000000
X.112	14	113	35	27	8	0.7714286	0.7714286
X.113	14	114	22	19	3	0.8636364	0.8636364
X.114	14	115	7	4	3	0.5714286	0.5714286
X.115	14	116	1	1	0	1.0000000	1.0000000
X.116	14	117	2	0	2	0.0000000	0.0000000
X.117	14	118	7	4	3	0.5714286	0.5714286
X.118	14	119	30	26	4	0.8666667	0.8666667
X.119	14	120	9	7	2	0.7777778	0.7777778
X.120	14	121	9	6	3	0.6666667	0.6666667
X.121	14	122	6	4	2	0.6666667	0.6666667
X.122	14	123	3	2	1	0.6666667	0.6666667
X.123	14	124	15	11	4	0.7333333	0.7333333
X.124	14	125	10	3	7	0.3000000	0.3000000
X.125	14	126	35	32	3	0.9142857	0.9142857
X.126	14	127	9	8	1	0.8888889	0.8888889
X.127	14	128	9	7	2	0.7777778	0.7777778
X.128	14	129	7	5	2	0.7142857	0.7142857
X.129	14	130	5	5	0	1.0000000	1.0000000
X.130	14	131	8	3	5	0.3750000	0.3750000
X.131	15	132	10	7	3	0.7000000	0.7000000
X.132	15	133	9	9	0	1.0000000	1.0000000
X.133	15	134	3	2	1	0.6666667	0.6666667
X.134	15	135	12	6	6	0.5000000	0.5000000
X.135	15	136	3	2	1	0.6666667	0.6666667
X.136	15	137	10	7	3	0.7000000	0.7000000
X.137	15	138	4	4	0	1.0000000	1.0000000
X.138	15	139	6	4	2	0.6666667	0.6666667
X.139	15	140	26	22	4	0.8461538	0.8461538
X.140	15	141	25	22	3	0.8800000	0.8800000
X.141	15	142	2	2	0	1.0000000	1.0000000
X.142	15	143	6	1	5	0.1666667	0.1666667
X.143	16	144	15	14	1	0.9333333	0.9333333
X.144	16	145	14	10	4	0.7142857	0.7142857
X.145	16	146	4	2	2	0.5000000	0.5000000
X.146	16	147	9	8	1	0.8888889	0.8888889
X.147	16	148	2	2	0	1.0000000	1.0000000
X.148	16	149	1	1	0	1.0000000	1.0000000
X.149	16	150	4	3	1	0.7500000	0.7500000
X.150	16	151	8	6	2	0.7500000	0.7500000
X.151	16	152	30	21	9	0.7000000	0.7000000

X.152	16	153	24	20	4	0.8333333	0.8333333
X.153	17	154	12	10	2	0.8333333	0.8333333
X.154	17	155	2	2	0	1.0000000	1.0000000
X.155	17	156	6	1	5	0.1666667	0.1666667
X.156	17	157	11	10	1	0.9090909	0.9090909
X.157	17	158	14	11	3	0.7857143	0.7857143
X.158	17	159	11	6	5	0.5454545	0.5454545
X.159	17	160	20	16	4	0.8000000	0.8000000
X.160	18	161	41	35	6	0.8536585	0.8536585
X.161	18	162	16	13	3	0.8125000	0.8125000
X.162	18	163	4	3	1	0.7500000	0.7500000
X.163	19	164	15	13	2	0.8666667	0.8666667
X.164	20	165	18	16	2	0.8888889	0.8888889
X.165	20	166	4	3	1	0.7500000	0.7500000
X.166	20	167	12	10	2	0.8333333	0.8333333
X.167	20	168	10	8	2	0.8000000	0.8000000
X.168	20	169	2	2	0	1.0000000	1.0000000
X.169	20	170	10	7	3	0.7000000	0.7000000
X.170	20	171	12	10	2	0.8333333	0.8333333
X.171	21	172	4	4	0	1.0000000	1.0000000
X.172	22	173	14	12	2	0.8571429	0.8571429
X.173	22	174	10	7	3	0.7000000	0.7000000
X.174	23	175	1	1	0	1.0000000	1.0000000
X.175	24	176	7	7	0	1.0000000	1.0000000
X.176	24	177	9	7	2	0.7777778	0.7777778
X.177	24	178	14	10	4	0.7142857	0.7142857
X.178	25	179	19	14	5	0.7368421	0.7368421
X.179	25	180	6	4	2	0.6666667	0.6666667
X.180	25	181	6	2	4	0.3333333	0.3333333
X.181	25	182	9	7	2	0.7777778	0.7777778
X.182	25	183	20	13	7	0.6500000	0.6500000
X.183	25	184	19	17	2	0.8947368	0.8947368
X.184	25	185	7	7	0	1.0000000	1.0000000

	Lower	Upper
X	0.207659601	0.9829022
X.1	0.434354699	0.9025394
X.2	0.621180172	0.9626387
X.3	0.685312956	0.9963362
X.4	0.622641564	0.9953370
X.5	0.436497178	0.9914511
X.6	0.051293294	1.0000000
X.7	0.025646647	0.9743534
X.8	0.390622089	0.8618799
X.9	0.700854952	1.0000000
X.10	0.409275430	0.9285208
X.11	0.508323063	0.8508535
X.12	0.000000000	0.6576198
X.13	0.646120089	0.9957256
X.14	0.565517535	1.0000000
X.15	0.646120089	0.9957256
X.16	0.323695346	0.7165599
X.17	0.562246805	0.8345651
X.18	0.700854952	1.0000000
X.19	0.436497178	0.9914511

X.20	0.250458365	0.8417801
X.21	0.467694665	0.9110583
X.22	0.305742395	0.8631557
X.23	0.342380228	1.0000000
X.24	0.358934452	0.9177811
X.25	0.675592435	1.0000000
X.26	0.595849973	0.9948707
X.27	0.700854952	1.0000000
X.28	0.230724281	0.8823792
X.29	0.577653690	0.9567418
X.30	0.701834701	0.9965804
X.31	0.198244961	0.6425317
X.32	0.675592435	1.0000000
X.33	0.355228915	0.8229029
X.34	0.452588969	0.9367749
X.35	0.595849973	0.9948707
X.36	0.207659601	0.9829022
X.37	0.453734520	0.8280525
X.38	0.017097765	0.7923404
X.39	0.583982568	0.9193423
X.40	0.434354699	0.9025394
X.41	0.107791267	0.6032219
X.42	0.664688007	0.9384997
X.43	0.497436241	0.9182047
X.44	0.641469294	0.9332132
X.45	0.187616306	0.8123837
X.46	0.305742395	0.8631557
X.47	0.716712624	0.9967942
X.48	0.354202136	0.8794162
X.49	0.300641843	0.9871767
X.50	0.354202136	0.8794162
X.51	0.452588969	0.9367749
X.52	0.438502968	1.0000000
X.53	0.577653690	0.9567418
X.54	0.452588969	0.9367749
X.55	0.771904628	1.0000000
X.56	0.300641843	0.9871767
X.57	0.716712624	0.9967942
X.58	0.672002349	0.9689805
X.59	0.480495659	0.8910255
X.60	0.730179694	0.9969827
X.61	0.771904628	1.0000000
X.62	0.710240968	0.9599676
X.63	0.645669565	1.0000000
X.64	0.207659601	0.9829022
X.65	0.305742395	0.8631557
X.66	0.300641843	0.9871767
X.67	0.453509157	0.8827862
X.68	0.413003635	0.8269028
X.69	0.685312956	0.9963362
X.70	0.523019438	0.9486323
X.71	0.486872171	0.9926724
X.72	0.656636494	0.9671202
X.73	0.342380228	1.0000000

X.74	0.701834701	0.9965804
X.75	0.595849973	0.9948707
X.76	0.505016835	0.8981793
X.77	0.645669565	1.0000000
X.78	0.025646647	0.9743534
X.79	0.838058948	0.9983454
X.80	0.741167033	1.0000000
X.81	0.589705414	0.9380887
X.82	0.266651293	0.8112215
X.83	0.358934452	0.9177811
X.84	0.512084491	0.8819359
X.85	0.000000000	0.9487067
X.86	0.685312956	0.9963362
X.87	0.325906446	0.7861920
X.88	0.529111818	0.9935883
X.89	0.685312956	0.9963362
X.90	0.510109164	1.0000000
X.91	0.581511269	0.8365362
X.92	0.358934452	0.9177811
X.93	0.491273434	0.8750025
X.94	0.438502968	1.0000000
X.95	0.396778147	0.8922087
X.96	0.486872171	0.9926724
X.97	0.510109164	1.0000000
X.98	0.250458365	0.8417801
X.99	0.600586205	0.9599061
X.100	0.524107694	0.9242861
X.101	0.595849973	0.9948707
X.102	0.784689197	1.0000000
X.103	0.300641843	0.9871767
X.104	0.396778147	0.8922087
X.105	0.505016835	0.8981793
X.106	0.641469294	0.9332132
X.107	0.305742395	0.8631557
X.108	0.565600468	0.8987696
X.109	0.646120089	0.9957256
X.110	0.300641843	0.9871767
X.111	0.342380228	1.0000000
X.112	0.609826831	0.8793412
X.113	0.666650128	0.9525100
X.114	0.250458365	0.8417801
X.115	0.051293294	1.0000000
X.116	0.000000000	0.6576198
X.117	0.250458365	0.8417801
X.118	0.703186733	0.9469034
X.119	0.452588969	0.9367749
X.120	0.354202136	0.8794162
X.121	0.299993315	0.9032286
X.122	0.207659601	0.9829022
X.123	0.480495659	0.8910255
X.124	0.107791267	0.6032219
X.125	0.776207260	0.9704176
X.126	0.565000294	0.9943007
X.127	0.452588969	0.9367749

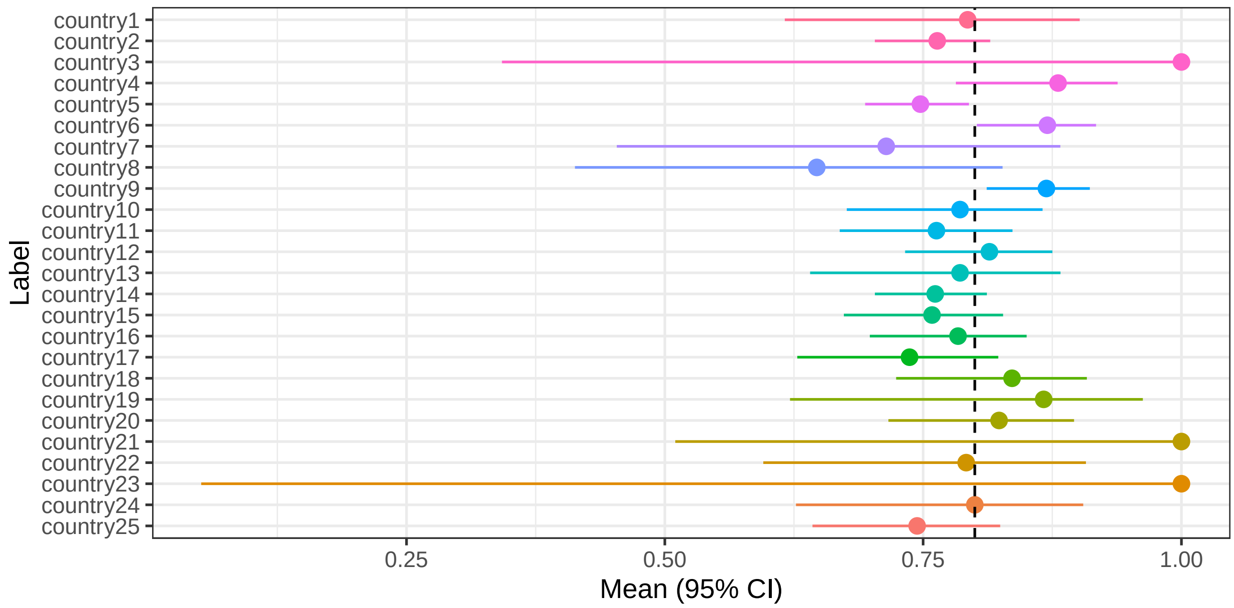
X.128	0.358934452	0.9177811
X.129	0.565517535	1.0000000
X.130	0.136844286	0.6942576
X.131	0.396778147	0.8922087
X.132	0.700854952	1.0000000
X.133	0.207659601	0.9829022
X.134	0.253781598	0.7462184
X.135	0.207659601	0.9829022
X.136	0.396778147	0.8922087
X.137	0.510109164	1.0000000
X.138	0.299993315	0.9032286
X.139	0.664688007	0.9384997
X.140	0.700442061	0.9583318
X.141	0.342380228	1.0000000
X.142	0.008548882	0.5635028
X.143	0.701834701	0.9965804
X.144	0.453509157	0.8827862
X.145	0.150038989	0.8499610
X.146	0.565000294	0.9943007
X.147	0.342380228	1.0000000
X.148	0.051293294	1.0000000
X.149	0.300641843	0.9871767
X.150	0.409275430	0.9285208
X.151	0.521242125	0.8333525
X.152	0.641469294	0.9332132
X.153	0.551969138	0.9530349
X.154	0.342380228	1.0000000
X.155	0.008548882	0.5635028
X.156	0.622641564	0.9953370
X.157	0.524107694	0.9242861
X.158	0.280091537	0.7872873
X.159	0.583982568	0.9193423
X.160	0.715565420	0.9311575
X.161	0.569911190	0.9340840
X.162	0.300641843	0.9871767
X.163	0.621180172	0.9626387
X.164	0.672002349	0.9689805
X.165	0.300641843	0.9871767
X.166	0.551969138	0.9530349
X.167	0.490162472	0.9433178
X.168	0.342380228	1.0000000
X.169	0.396778147	0.8922087
X.170	0.551969138	0.9530349
X.171	0.510109164	1.0000000
X.172	0.600586205	0.9599061
X.173	0.396778147	0.8922087
X.174	0.051293294	1.0000000
X.175	0.645669565	1.0000000
X.176	0.452588969	0.9367749
X.177	0.453509157	0.8827862
X.178	0.512084491	0.8819359
X.179	0.299993315	0.9032286
X.180	0.096771411	0.7000067
X.181	0.452588969	0.9367749

```
X.182 0.432854277 0.8188082
X.183 0.686059173 0.9706414
X.184 0.645669565 1.0000000
```

```
foox1 <- foo %>% group_by(country) %>% summarise(N = length(y),
    ones = (mean(y)) * length(y), zero = (1 - mean(y)) * length(y),
    mean = mean(y), )

foox1 <- data.frame(foox1)
names(foox1) <- c("Country", "N", "Yes", "No", "Mean")
```

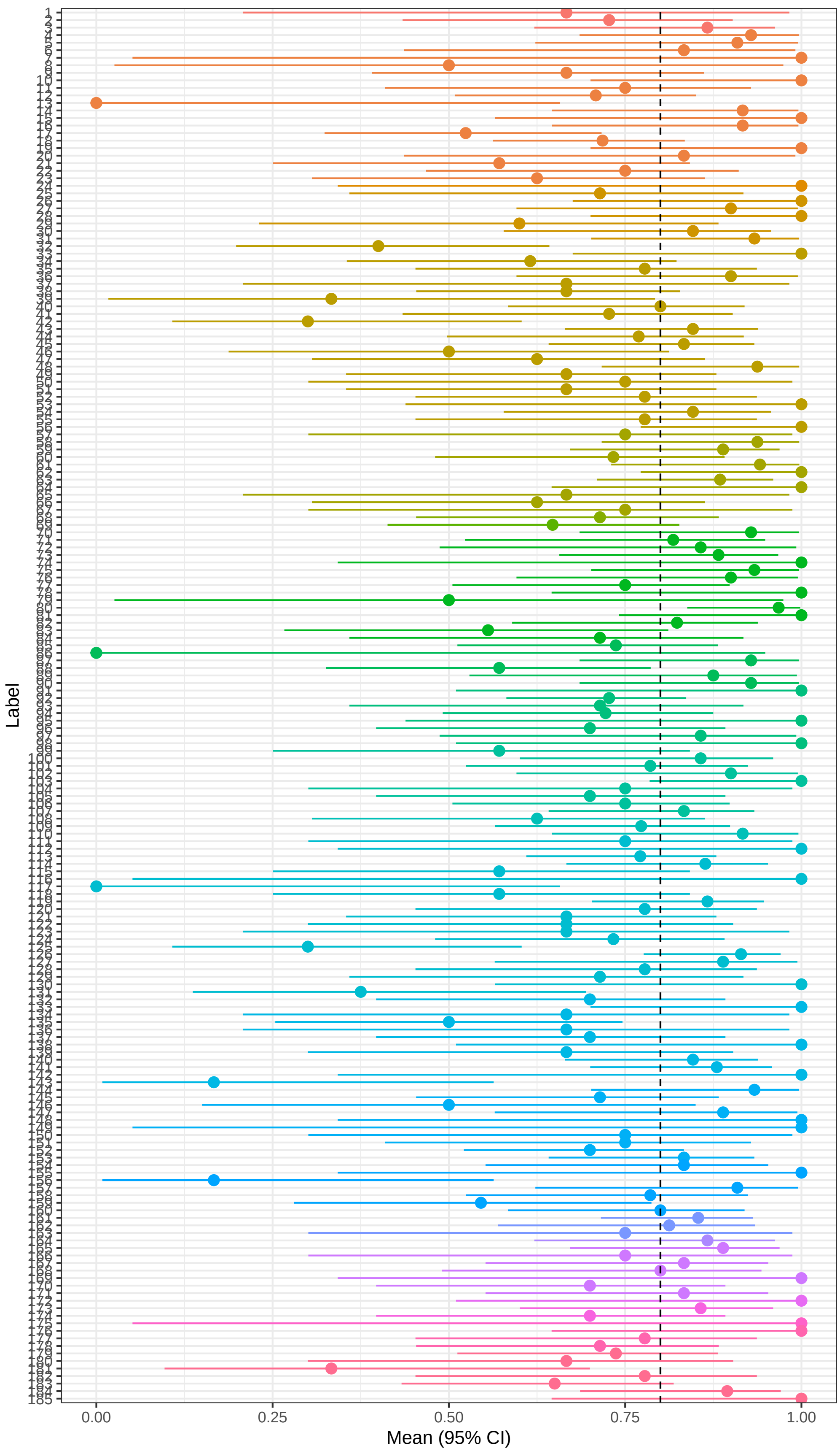
Plot crude estimates country level

```
foox1 <- cbind(foox1,binconf(foox1$Yes, foox1$N))
est <- foox1

label <- paste0("country", 1:25)
mean  <-  est[,6]
lower <-  est[,7]
upper <-  est[,8]

df <- data.frame(label, mean, lower, upper)

# reverses the factor level ordering for labels after coord_flip()
df$label <- factor(df$label, levels=rev(df$label))


library(ggplot2)

fp <- NULL
fp <- ggplot(data=df, aes(x=label, y=mean, ymin=lower, ymax=upper, colour=label)) +
        geom_pointrange() +
        geom_hline(yintercept=mu, lty=2) +  # add a dotted line at x=1 after flip
        coord_flip() +  # flip coordinates (puts labels on y axis)
        xlab("Label") + ylab("Mean (95% CI)") +
        theme_bw() +   # use a white background
        theme(legend.position="none")
print(fp)
```



Plot crude site estimates

```
est <- foox
label <-  est$site
mean  <-  est[,6]
lower <-  est[,8]
upper <-  est[,9]


df <- data.frame(co=foox$Country, label, mean, lower, upper)
#df <- df[order(label),]

# reverses the factor level ordering for labels after coord_flip()
df$label <- factor(df$label, levels=rev(df$label))

fp <- NULL
fp <- ggplot(data=df, aes(x=label, y=mean, ymin=lower, ymax=upper, colour=co)) +
        geom_pointrange() +
        geom_hline(yintercept=mu, lty=2) +  # add a dotted line at x=1 after flip
        coord_flip() +  # flip coordinates (puts labels on y axis)
        xlab("Label") + ylab("Mean (95% CI)") +
        theme_bw() + # use a white background +
  theme(legend.position="none")

print(fp)
```


Label
0.00
0.25
0.50
0.75
1.00
Mean (95% CI)

```
# cat("\n")
# #cat("Summary Statistics")
# cat("\n")
# print(kable(foox, format="pandoc", digits=c(0,0,0,4),
#             caption = "crude estimates"))
# cat("\n")
```

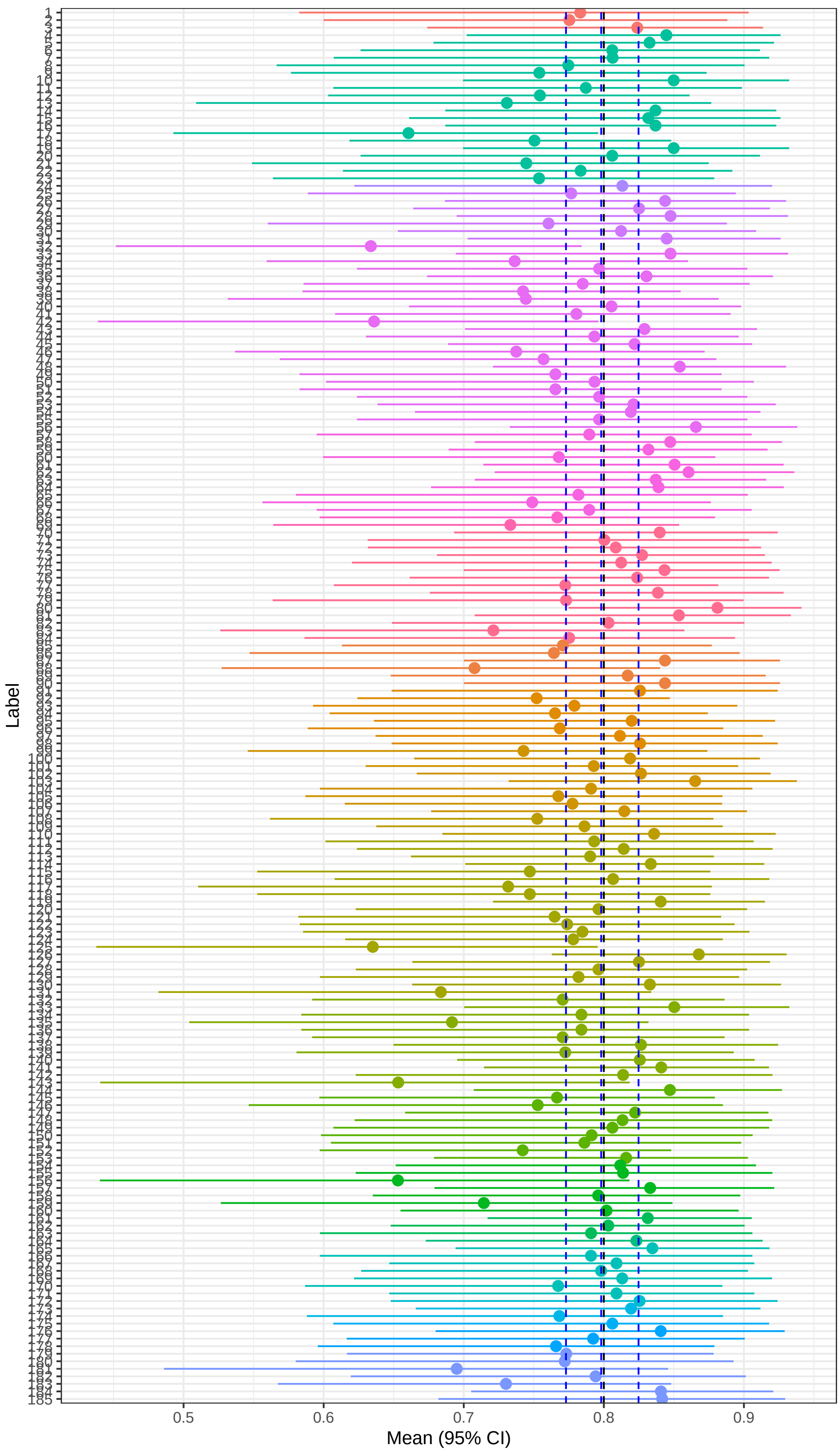
lmer estimates sites

```
# manage the data
# fco <- coef(fit0)$`site:country`
# c(attr(ranef(fit0,condVar=TRUE)[[1]],"postVar"))  # co
str(rr1 <- ranef(fit0, condVar = TRUE))
```

```
List of 2
 $ site:country:'data.frame':   185 obs. of  1 variable:
  ..$ (Intercept): num [1:185] -0.0902 -0.1357 0.1682 0.3187 0.2298 ...
  ..- attr(*, "postVar")= num [1, 1, 1:185] 0.229 0.175 0.167 0.176 0.185 ...
 $ country     :'data.frame':   25 obs. of  1 variable:
  ..$ (Intercept): num [1:25] -0.00302 -0.02918 0.00504 0.04673 -0.08992 ...
  ..- attr(*, "postVar")= num [1, 1, 1:25] 0.013 0.01043 0.0135 0.01239 0.00957 ...
 - attr(*, "class")= chr "ranef.mer"
```

```
# dotplot(rr1)                    ## default  #
# cV <- ranef(fit0, condVar = TRUE)
# # ranvar <- attr(cV[[1]], "postVar")
# # sqrt(diag(ranvar[,,]))

# get random effects from lmer and calculate confidence intervals too
# http://bbolker.github.io/mixedmodels-misc/glmmFAQ.html#lme4
# intercept
fix <- fixef(fit0)              # intercept on log scale
fix.var <- sqrt(diag(vcov(fit0)))   # intercept associated standard error^2
# blups
s.c <- rr1[1][1]                # site country random effects
s.c.var <- c(attr(ranef(fit0,condVar=TRUE)[[1]],"postVar"))^0.5  # site country sds
# make a datset
vars <- as.data.frame(cbind(intercept=fix, intercept.var=fix.var ,
                            blup=as.vector(unlist(s.c)), blup.var=s.c.var ))
# calculate CI for the random effects
vars$est <- vars$intercept+vars$blup  # shift from intercept
vars$lower <- vars$est+c(-1)* qnorm(0.975)*sqrt(vars$blup.var^2+vars$intercept.var^2)  #lower CI
vars$upper <- vars$est+c( 1)* qnorm(0.975)*sqrt(vars$blup.var^2+vars$intercept.var^2)  #upper CI

#log odds to probabilities
A <- function(x) 1/(1+exp(-x))
df <- vars
df <- data.frame(df[,c(1:4)], apply(df[,c(5:7)],2, A) )


label = rownames(s.c$`site:country`)
df$sites <-     gsub(":.*","", label)
df$countries <- gsub(".*:","", label)
```

```
#plot
est <- df
label <-  est$sites
mean  <-  est$est
lower <-  est$lower
upper <-  est$upper

# intercept CI
fit0fci <- confint(fit0)
L <- fit0fci["(Intercept)",][1][[1]]
U <- fit0fci["(Intercept)",][2][[1]]

df <- data.frame(co=df$countries, label, mean, lower, upper)
#df <- df[order(label),]

# reverses the factor level ordering for labels after coord_flip()
df$label <- factor(df$label, levels=rev(df$label))

fp <- NULL
fp <- ggplot(data=df, aes(x=label, y=mean, ymin=lower, ymax=upper, colour=co)) +
        geom_pointrange() +
        geom_hline(yintercept=mu, lty=2) +  # add a dotted line at x=1 after flip
        geom_hline(yintercept= A(fix),  color="blue", linetype="dashed") +  # estimate
        geom_hline(yintercept=A(L),  color="blue", linetype="dashed") +
        geom_hline(yintercept=A(U),  color="blue", linetype="dashed") +
        coord_flip() +  # flip coordinates (puts labels on y axis)
        xlab("Label") + ylab("Mean (95% CI)") +
        theme_bw() + # use a white background +
  theme(legend.position="none")

print(fp)
```


Label
Mean (95% CI)
0.5
0.6
0.7
0.8
0.9

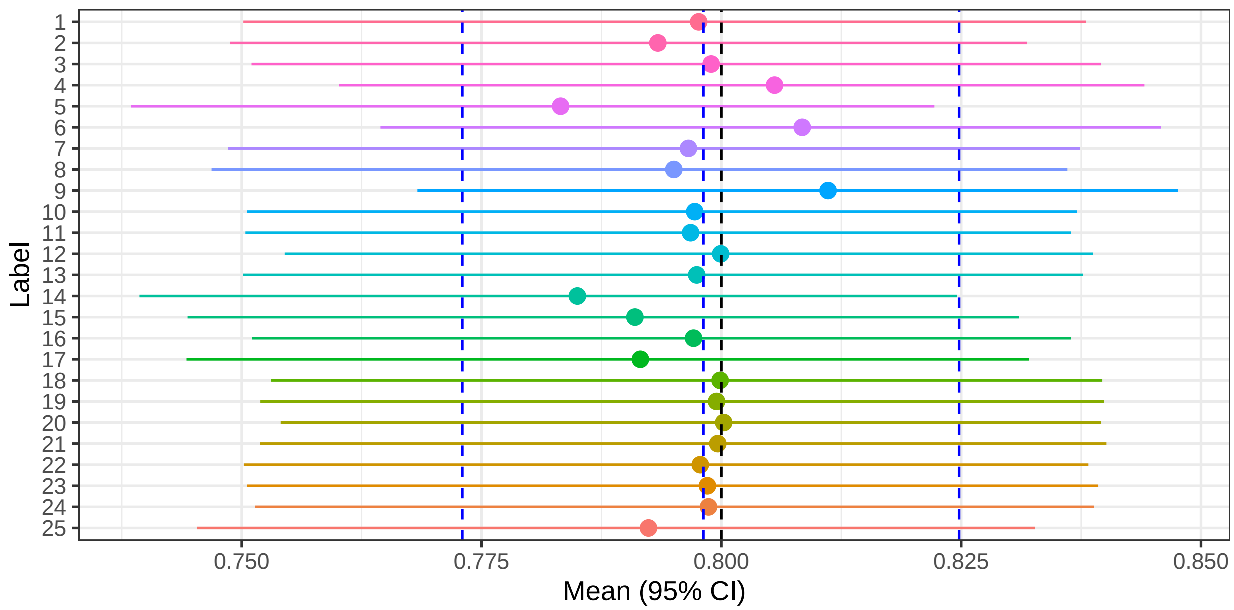
lmer estimates countries

```
str(rr1 <- ranef(fit0, condVar = TRUE))
```

```
List of 2
 $ site:country:'data.frame':   185 obs. of  1 variable:
  ..$ (Intercept): num [1:185] -0.0902 -0.1357 0.1682 0.3187 0.2298 ...
  ..- attr(*, "postVar")= num [1, 1, 1:185] 0.229 0.175 0.167 0.176 0.185 ...
 $ country      :'data.frame':   25 obs. of  1 variable:
  ..$ (Intercept): num [1:25] -0.00302 -0.02918 0.00504 0.04673 -0.08992 ...
  ..- attr(*, "postVar")= num [1, 1, 1:25] 0.013 0.01043 0.0135 0.01239 0.00957 ...
 - attr(*, "class")= chr "ranef.mer"
```

```
  # get random effects from lmer and calculate confidence intervals too
  # http://bbolker.github.io/mixedmodels-misc/glmmFAQ.html#lme4
  # blups
  s.c <- rr1[2][1]                # country random effects
  s.c.var <- c(attr(ranef(fit0,condVar=TRUE)[[2]],"postVar"))^0.5  # country sds
  # make a datset
  vars <- as.data.frame(cbind(intercept=fix, intercept.var=fix.var ,
                            blup=as.vector(unlist(s.c)), blup.var=s.c.var ))
  # calculate CI for the random effects
  vars$est <- vars$intercept+vars$blup  # shift from intercept
  vars$lower <- vars$est+c(-1)* qnorm(0.975)*sqrt(vars$blup.var^2+vars$intercept.var^2)  #lower CI
  vars$upper <- vars$est+c( 1)* qnorm(0.975)*sqrt(vars$blup.var^2+vars$intercept.var^2)  #upper CI

  #log odds to probabilities
  A <- function(x) 1/(1+exp(-x))
  df <- vars
  df <- data.frame(df[,c(1:4)], apply(df[,c(5:7)],2, A) )


  label = rownames(s.c$`country`)
  df$countries <- gsub(".*:","", label)

  #plot
  est <- df
  label <-  est$countries
  mean  <-  est$est
  lower <-  est$lower
  upper <-  est$upper

  # intercept CI
# fit0fci <- confint(fit0)
#  L <- fit0fci["(Intercept)",][1][[1]]
#  U <- fit0fci["(Intercept)",][2][[1]]

  df <- data.frame( label, mean, lower, upper)
  #df <- df[order(label),]

  # reverses the factor level ordering for labels after coord_flip()
  df$label <- factor(df$label, levels=rev(df$label))

  fp <- NULL
  fp <- ggplot(data=df, aes(x=label, y=mean, ymin=lower, ymax=upper, colour=label)) +
```

```
        geom_pointrange() +
        geom_hline(yintercept=mu, lty=2) +  # add a dotted line at x=1 after flip
        geom_hline(yintercept= A(fix),   color="blue", linetype="dashed") +  # estimate
        geom_hline(yintercept=A(L),   color="blue", linetype="dashed") +
        geom_hline(yintercept=A(U),  color="blue", linetype="dashed") +
        coord_flip() +  # flip coordinates (puts labels on y axis)
        xlab("Label") + ylab("Mean (95% CI)") +
        theme_bw() + # use a white background +
  theme(legend.position="none")

print(fp)
```



REFERENCES

1 paper http://bmcmedresmethodol.biomedcentral.com/track/pdf/10.1186/1471-2288-11-94?site=bmcmedresmethodol.biomedcentral.com 1 code http://www.biomedcentral.com/content/supplementary/1471-2288-11-94-S1.PDF 1 code https://bmcmedresmethodol.biomedcentral.com/articles/10.1186/1471-2288-11-94#MOESM1 2 forrest plot https://stackoverflow.com/questions/38062650/forest-plot-for-a-beginner-simple-exa
3 https://stackoverflow.com/questions/14639892/how-to-extract-words-between-two-period-using-rs-gsub
4 https://stackoverflow.com/questions/31774086/extracting-text-after-last-period-in-string-in-r 5 https://stats.stackexchange.com/questions/147836/prediction-interval-for-lmer-mixed-effects-model-in-r 6 http://bbolker.github.io/mixedmodels-misc/glmmFAQ.html#lme4

COMPUTING ENVIRONMENT

```
R version 3.6.1 (2019-07-05)
Platform: x86_64-w64-mingw32/x64 (64-bit)
Running under: Windows 10 x64 (build 17134)

Matrix products: default

locale:
[1] LC_COLLATE=English_United Kingdom.1252
[2] LC_CTYPE=English_United Kingdom.1252
[3] LC_MONETARY=English_United Kingdom.1252
[4] LC_NUMERIC=C
[5] LC_TIME=English_United Kingdom.1252

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods
[7] base

other attached packages:
 [1] dplyr_0.8.3          rstan_2.19.2         StanHeaders_2.18.1-10
 [4] brms_2.9.0           Rcpp_1.0.1           rms_5.1-3.1
 [7] SparseM_1.77         Hmisc_4.2-0          ggplot2_3.2.0
[10] Formula_1.2-3        survival_2.44-1.1    lattice_0.20-38
[13] lme4_1.1-21          Matrix_1.2-17        knitr_1.23

loaded via a namespace (and not attached):
  [1] TH.data_1.0-10      minqa_1.2.4         colorspace_1.4-1
  [4] ggridges_0.5.1      rsconnect_0.8.13    htmlTable_1.13.1
  [7] markdown_1.0        base64enc_0.1-3     rstudioapi_0.10
 [10] MatrixModels_0.4-1  DT_0.7              fansi_0.4.0
 [13] mvtnorm_1.0-11      bridgesampling_0.6-0 codetools_0.2-16
 [16] splines_3.6.1       shinythemes_1.1.2   zeallot_0.1.0
 [19] bayesplot_1.7.0     nloptr_1.2.1        binom_1.1-1
 [22] cluster_2.1.0       shiny_1.3.2         compiler_3.6.1
 [25] backports_1.1.4     assertthat_0.2.1    lazyeval_0.2.2
 [28] cli_1.1.0           formatR_1.7         later_0.8.0
 [31] prettyunits_1.0.2   acepack_1.4.1       htmltools_0.3.6
 [34] quantreg_5.41       tools_3.6.1         igraph_1.2.4.1
 [37] coda_0.19-3         gtable_0.3.0        glue_1.3.1
 [40] reshape2_1.4.3      vctrs_0.2.0         nlme_3.1-140
 [43] crosstalk_1.0.0     xfun_0.8            stringr_1.4.0
 [46] ps_1.3.0            mime_0.7            miniUI_0.1.1.1
 [49] gtools_3.8.1        polspline_1.1.15    MASS_7.3-51.4
 [52] zoo_1.8-6           scales_1.0.0        colourpicker_1.0
 [55] promises_1.0.1      Brobdingnag_1.2-6   parallel_3.6.1
 [58] sandwich_2.5-1      inline_0.3.15       shinystan_2.5.0
 [61] RColorBrewer_1.1-2  yaml_2.2.0          gridExtra_2.3
 [64] loo_2.1.0           rpart_4.1-15        latticeExtra_0.6-28
 [67] stringi_1.4.3       dygraphs_1.1.1.6    checkmate_1.9.4
 [70] boot_1.3-22         pkgbuild_1.0.3      rlang_0.4.0
 [73] pkgconfig_2.0.2     matrixStats_0.54.0  evaluate_0.14
 [76] purrr_0.3.2         labeling_0.3        rstantools_1.5.1
 [79] htmlwidgets_1.3     tidyselect_0.2.5    processx_3.4.0
```

```
 [82] plyr_1.8.4          magrittr_1.5        R6_2.4.0
 [85] multcomp_1.4-10     pillar_1.4.2        foreign_0.8-71
 [88] withr_2.1.2         xts_0.11-2          abind_1.4-5
 [91] nnet_7.3-12         tibble_2.1.3        crayon_1.3.4
 [94] utf8_1.1.4          rmarkdown_1.14      grid_3.6.1
 [97] data.table_1.12.2   callr_3.3.0         threejs_0.3.1
[100] digest_0.6.20       xtable_1.8-4        httpuv_1.5.1
[103] stats4_3.6.1        munsell_0.5.0       viridisLite_0.3.0
[106] shinyjs_1.0
```

```
[1] "C:/Users/eam2018/Documents/hierarchical-binomial-random-effects"
```

This took 382.11 seconds to execute.

```
[1] FALSE
```