Computational Statistics & Probability

Problem Set 5 - Multilevel Models

Author: Neelesh Bhalla
Collaborators: Nils Marthiensen, Chia-Jung Chang

2022-12-15

1. Varying Slopes and Effective Parameters

When is it possible for a varying slopes model to have fewer effective parameters (as estimated by WAIC or PSIS) than the corresponding model with fixed slopes? Explain your answer.

```
# Consider a case of tight priors. If the prior assigned to each intercept shrinks
# them all towards the mean, this will result in fewer effective parameters.

# If we have an aggressive regularizing prior, this will result in a less
# flexible posterior and therefore fewer effective parameters

# When there is little or next-to-no variation among clusters.
# The absence of this among-cluster variation induces very strong shrinkage.
# As a result, albeit containing more actual parameters in the posterior
# distribution, the varying slopes model may end up less flexible in fitting
# to the data because of adaptive regularization forcing strong shrinkage.
# Consequently, our number of effective parameters - a proxy of over-fitting
# risk and posterior flexibility - decreases.
```

```
# For demonstration, we can consult the comparison of models m13.1 and m13.2 in R Code 13.4 in the book

# The models are applied on Reed frog tadpole mortality data.

# We are interested in number surviving, out of an initial count of tadpoles.

library(rethinking)
```

```
## Loading required package: rstan
```

```
## Loading required package: StanHeaders
```

```
##
## rstan version 2.26.13 (Stan version 2.26.1)
```

```
## For execution on a local, multicore CPU with excess RAM we recommend calling
## options(mc.cores = parallel::detectCores()).
## To avoid recompilation of unchanged Stan programs, we recommend calling
## rstan_options(auto_write = TRUE)
## For within-chain threading using `reduce_sum()` or `map_rect()` Stan functions,
## change `threads_per_chain` option:
## rstan_options(threads_per_chain = 1)
```

```
## Loading required package: cmdstanr
```

```
## This is cmdstanr version 0.5.3
## - CmdStanR documentation and vignettes: mc-stan.org/cmdstanr
## - CmdStan path: /Users/neelesh/.cmdstan/cmdstan-2.30.1
## - CmdStan version: 2.30.1
##
## A newer version of CmdStan is available. See ?install_cmdstan() to install it.
## To disable this check set option or environment variable CMDSTANR_NO_VER_CHECK=TRUE.
## Loading required package: parallel
## rethinking (Version 2.21)
##
## Attaching package: 'rethinking'
## The following object is masked from 'package:rstan':
##
##     stan
## The following object is masked from 'package:stats':
##
##     rstudent
```

```
data(reedfrogs)
d <- reedfrogs
str(d)
```

```
## 'data.frame':    48 obs. of  5 variables:
##  $ density : int  10 10 10 10 10 10 10 10 10 10 ...
##  $ pred    : Factor w/ 2 levels "no","pred": 1 1 1 1 1 1 1 1 2 2 ...
##  $ size    : Factor w/ 2 levels "big","small": 1 1 1 1 2 2 2 2 1 1 ...
##  $ surv    : int  9 10 7 10 9 9 10 9 4 9 ...
##  $ propsurv: num  0.9 1 0.7 1 0.9 0.9 1 0.9 0.4 0.9 ...
```

```
d$tank <- 1:nrow(d)
dat <- list(
S = d$surv,
N = d$density,
tank = d$tank )

# approximate posterior for ordinary fixed model
m13.1 <- ulam(
  alist(
    S ~ dbinom( N , p ) ,
    logit(p) <- a[tank] ,
    a[tank] ~ dnorm( 0 , 1.5 )
  ), data=dat , chains=4 , log_lik=TRUE, refresh=0 )
```

```
## Warning in '/var/folders/wx/1_76tj0s15gc4yxmndvw4l_00000gn/T/RtmpLDdUPF/model-1262f632ec6c0.stan', li
##     of arrays by placing brackets after a variable name is deprecated and
##     will be removed in Stan 2.32.0. Instead use the array keyword before the
##     type. This can be changed automatically using the auto-format flag to
##     stanc
## Warning in '/var/folders/wx/1_76tj0s15gc4yxmndvw4l_00000gn/T/RtmpLDdUPF/model-1262f632ec6c0.stan', li
##     of arrays by placing brackets after a variable name is deprecated and
##     will be removed in Stan 2.32.0. Instead use the array keyword before the
```

```
##     type. This can be changed automatically using the auto-format flag to
##     stanc
## Warning in '/var/folders/wx/1_76tj0s15gc4yxmndvw4l_00000gn/T/RtmpLDdUPF/model-1262f632ec6c0.stan', li
##     of arrays by placing brackets after a variable name is deprecated and
##     will be removed in Stan 2.32.0. Instead use the array keyword before the
##     type. This can be changed automatically using the auto-format flag to
##     stanc

## Running MCMC with 4 sequential chains, with 1 thread(s) per chain...
##
## Chain 1 finished in 0.2 seconds.
## Chain 2 finished in 0.3 seconds.
## Chain 3 finished in 0.3 seconds.
## Chain 4 finished in 0.2 seconds.
##
## All 4 chains finished successfully.
## Mean chain execution time: 0.2 seconds.
## Total execution time: 1.5 seconds.
```

```
# the multilevel model
m13.2 <- ulam(
  alist(
    S ~ dbinom( N , p ) ,
    logit(p) <- a[tank] ,
    a[tank] ~ dnorm( a_bar , sigma ) ,
    a_bar ~ dnorm( 0 , 1.5 ) ,
    sigma ~ dexp( 1 )
  ), data=dat , chains=4 , log_lik=TRUE, refresh=0 )
```

```
## Warning in '/var/folders/wx/1_76tj0s15gc4yxmndvw4l_00000gn/T/RtmpLDdUPF/model-1262f6412a793.stan', li
##     of arrays by placing brackets after a variable name is deprecated and
##     will be removed in Stan 2.32.0. Instead use the array keyword before the
##     type. This can be changed automatically using the auto-format flag to
##     stanc
## Warning in '/var/folders/wx/1_76tj0s15gc4yxmndvw4l_00000gn/T/RtmpLDdUPF/model-1262f6412a793.stan', li
##     of arrays by placing brackets after a variable name is deprecated and
##     will be removed in Stan 2.32.0. Instead use the array keyword before the
##     type. This can be changed automatically using the auto-format flag to
##     stanc
## Warning in '/var/folders/wx/1_76tj0s15gc4yxmndvw4l_00000gn/T/RtmpLDdUPF/model-1262f6412a793.stan', li
##     of arrays by placing brackets after a variable name is deprecated and
##     will be removed in Stan 2.32.0. Instead use the array keyword before the
##     type. This can be changed automatically using the auto-format flag to
##     stanc

## Running MCMC with 4 sequential chains, with 1 thread(s) per chain...
##
## Chain 1 finished in 0.2 seconds.
## Chain 2 finished in 0.2 seconds.

## Chain 3 Informational Message: The current Metropolis proposal is about to be rejected because of the

## Chain 3 Exception: normal_lpdf: Scale parameter is 0, but must be positive! (in '/var/folders/wx/1_76

## Chain 3 If this warning occurs sporadically, such as for highly constrained variable types like cova

## Chain 3 but if this warning occurs often then your model may be either severely ill-conditioned or mi
```

```
## Chain 3
```

```
## Chain 3 finished in 0.2 seconds.
## Chain 4 finished in 0.2 seconds.
##
## All 4 chains finished successfully.
## Mean chain execution time: 0.2 seconds.
## Total execution time: 1.3 seconds.
```

```
# comparing the models
compare( m13.1 , m13.2 )
```

```
##           WAIC       SE    dWAIC      dSE    pWAIC       weight
## m13.2 200.4765 7.429928  0.00000       NA 21.11060 0.9991284751
## m13.1 214.5652 4.766827 14.08879 4.015452 25.66746 0.0008715249
```

```
# The multilevel model has only 21 effective parameters. There are 28 fewer effective
# parameters than actual parameters, because the prior assigned to each intercept
# shrinks them all towards the mean ¯. In this case, the prior is reasonably strong.
# the amount of regularization has been learned from the data itself

# The multilevel model m13.2 has fewer effective parameters than the ordinary fixed
# model m13.1. This is despite the fact that the ordinary model has fewer actual
# parameters, only 48 (i.e. the number of observations in the data) instead
# of 50 with m13.2 (one overall sample intercept ¯, the standard deviation among tanks ,
# and then 48 per-tank intercepts..)

# The extra two parameters in the multilevel
# model allowed it to learn a more aggressive regularizing prior, to adaptively regularize.
# This resulted in a less flexible posterior and therefore fewer effective parameters.

# This is explained in the literature as well.
```

2. Gaussian Process Regression

a) Go to section §14.5 in the textbook and compare the Gaussian process model of Oceanic tools, m14.8, to all the models fit to the same data in §11.2 by WAIC. This first step asks you to just produce the table.

```
# load the data
library(rethinking)
data(Kline2)
d <- Kline2
d
```

```
##       culture population contact total_tools mean_TU   lat    lon  lon2
## 1    Malekula       1100     low          13     3.2 -16.3  167.5 -12.5
## 2     Tikopia       1500     low          22     4.7 -12.3  168.8 -11.2
## 3  Santa Cruz       3600     low          24     4.0 -10.7  166.0 -14.0
## 4         Yap       4791    high          43     5.0   9.5  138.1 -41.9
## 5    Lau Fiji       7400    high          33     5.0 -17.7  178.1  -1.9
## 6   Trobriand       8000    high          19     4.0  -8.7  150.9 -29.1
## 7       Chuuk       9200    high          40     3.8   7.4  151.6 -28.4
## 8       Manus      13000     low          28     6.6  -2.1  146.9 -33.1
## 9       Tonga      17500    high          55     5.4 -21.2 -175.2   4.8
## 10     Hawaii     275000     low          71     6.6  19.9 -155.6  24.4
##       logpop
```

```
## 1   7.003065
## 2   7.313220
## 3   8.188689
## 4   8.474494
## 5   8.909235
## 6   8.987197
## 7   9.126959
## 8   9.472705
## 9   9.769956
## 10 12.524526
```

```
# Revisiting Gaussian process model (m14.8) of Oceanic tools from section §14.5

d$society <- 1:10 # index observations
data(islandsDistMatrix)

dat_list <- list(
  T = d$total_tools,
  P = d$population,
  society = d$society,
  Dmat=islandsDistMatrix )


m14.8 <- ulam(
  alist(
    T ~ dpois(lambda),
    lambda <- (a*P^b/g)*exp(k[society]),
    vector[10]:k ~ multi_normal( 0 , SIGMA ),
    matrix[10,10]:SIGMA <- cov_GPL2( Dmat , etasq , rhosq , 0.01 ),
    c(a,b,g) ~ dexp( 1 ),
    etasq ~ dexp( 2 ),
    rhosq ~ dexp( 0.5 )
  ), data=dat_list , chains=4 , cores=4 , iter=2000 , log_lik=TRUE, refresh=0)
```

```
## Warning in '/var/folders/wx/1_76tj0s15gc4yxmndvw4l_00000gn/T/RtmpLDdUPF/model-1262f2e96c208.stan', li
##     of arrays by placing brackets after a variable name is deprecated and
##     will be removed in Stan 2.32.0. Instead use the array keyword before the
##     type. This can be changed automatically using the auto-format flag to
##     stanc
## Warning in '/var/folders/wx/1_76tj0s15gc4yxmndvw4l_00000gn/T/RtmpLDdUPF/model-1262f2e96c208.stan', li
##     of arrays by placing brackets after a variable name is deprecated and
##     will be removed in Stan 2.32.0. Instead use the array keyword before the
##     type. This can be changed automatically using the auto-format flag to
##     stanc
## Warning in '/var/folders/wx/1_76tj0s15gc4yxmndvw4l_00000gn/T/RtmpLDdUPF/model-1262f2e96c208.stan', li
##     of arrays by placing brackets after a variable name is deprecated and
##     will be removed in Stan 2.32.0. Instead use the array keyword before the
##     type. This can be changed automatically using the auto-format flag to
##     stanc

## Running MCMC with 4 parallel chains, with 1 thread(s) per chain...

## Chain 1 Informational Message: The current Metropolis proposal is about to be rejected because of the

## Chain 1 Exception: multi_normal_lpdf: Covariance matrix is not symmetric. Covariance matrix[1,2] = na

## Chain 1 If this warning occurs sporadically, such as for highly constrained variable types like covar
```

```
## Chain 1 but if this warning occurs often then your model may be either severely ill-conditioned or mi
## Chain 1
## Chain 4 Informational Message: The current Metropolis proposal is about to be rejected because of the
## Chain 4 Exception: multi_normal_lpdf: Covariance matrix is not symmetric. Covariance matrix[1,2] = na
## Chain 4 If this warning occurs sporadically, such as for highly constrained variable types like covar
## Chain 4 but if this warning occurs often then your model may be either severely ill-conditioned or mi
## Chain 4
## Chain 2 finished in 10.0 seconds.
## Chain 3 finished in 10.2 seconds.
## Chain 4 finished in 10.2 seconds.
## Chain 1 finished in 10.4 seconds.
##
## All 4 chains finished successfully.
## Mean chain execution time: 10.2 seconds.
## Total execution time: 10.5 seconds.
```

```
# we set the ulam() argument log_lik=TRUE for comparison with WAIC in the next step.
```

```
# Revisiting models fit over same data from section §11.2

d$P <- scale( log(d$population) )
d$contact_id <- ifelse( d$contact=="high" , 2 , 1 )

dat <- list(
T = d$total_tools ,
P = d$P ,
cid = d$contact_id )


# intercept only
m11.9 <- ulam(
  alist(
    T ~ dpois( lambda ),
    log(lambda) <- a,
    a ~ dnorm(3,0.5)
  ), data=dat , chains=4 , log_lik=TRUE, refresh=0 )
```

```
## Warning in '/var/folders/wx/1_76tj0s15gc4yxmndvw4l_00000gn/T/RtmpLDdUPF/model-1262f5d9c49d5.stan', li
##     of arrays by placing brackets after a variable name is deprecated and
##     will be removed in Stan 2.32.0. Instead use the array keyword before the
##     type. This can be changed automatically using the auto-format flag to
##     stanc
## Warning in '/var/folders/wx/1_76tj0s15gc4yxmndvw4l_00000gn/T/RtmpLDdUPF/model-1262f5d9c49d5.stan', li
##     of arrays by placing brackets after a variable name is deprecated and
##     will be removed in Stan 2.32.0. Instead use the array keyword before the
##     type. This can be changed automatically using the auto-format flag to
##     stanc
## Running MCMC with 4 sequential chains, with 1 thread(s) per chain...
##
## Chain 1 finished in 0.0 seconds.
## Chain 2 finished in 0.0 seconds.
```

```
## Chain 3 finished in 0.0 seconds.
## Chain 4 finished in 0.0 seconds.
##
## All 4 chains finished successfully.
## Mean chain execution time: 0.0 seconds.
## Total execution time: 0.5 seconds.
```

```
# interaction model
m11.10 <- ulam(
  alist(
    T ~ dpois( lambda ),
    log(lambda) <- a[cid] + b[cid]*P,
    a[cid] ~ dnorm( 3 , 0.5 ),
    b[cid] ~ dnorm( 0 , 0.2 )
  ), data=dat , chains=4 , log_lik=TRUE, refresh=0 )
```

```
## Warning in '/var/folders/wx/1_76tj0s15gc4yxmndvw4l_00000gn/T/RtmpLDdUPF/model-1262f763557c6.stan', li
##     of arrays by placing brackets after a variable name is deprecated and
##     will be removed in Stan 2.32.0. Instead use the array keyword before the
##     type. This can be changed automatically using the auto-format flag to
##     stanc
## Warning in '/var/folders/wx/1_76tj0s15gc4yxmndvw4l_00000gn/T/RtmpLDdUPF/model-1262f763557c6.stan', li
##     of arrays by placing brackets after a variable name is deprecated and
##     will be removed in Stan 2.32.0. Instead use the array keyword before the
##     type. This can be changed automatically using the auto-format flag to
##     stanc
```

```
## Running MCMC with 4 sequential chains, with 1 thread(s) per chain...
##
## Chain 1 finished in 0.1 seconds.
## Chain 2 finished in 0.1 seconds.
## Chain 3 finished in 0.1 seconds.
## Chain 4 finished in 0.1 seconds.
##
## All 4 chains finished successfully.
## Mean chain execution time: 0.1 seconds.
## Total execution time: 0.9 seconds.
```

```
# the scientific model (model with tool innovation; under 'overthinking' section in the book)
dat2 <- list( T=d$total_tools, P=d$population, cid=d$contact_id )
m11.11 <- ulam(
  alist(
    T ~ dpois( lambda ),
    lambda <- exp(a[cid])*P^b[cid]/g,
    a[cid] ~ dnorm(1,1),
    b[cid] ~ dexp(1),
    g ~ dexp(1)
  ), data=dat2 , chains=4 , log_lik=TRUE, refresh=0 )
```

```
## Warning in '/var/folders/wx/1_76tj0s15gc4yxmndvw4l_00000gn/T/RtmpLDdUPF/model-1262f783f7adb.stan', li
##     of arrays by placing brackets after a variable name is deprecated and
##     will be removed in Stan 2.32.0. Instead use the array keyword before the
##     type. This can be changed automatically using the auto-format flag to
##     stanc
## Warning in '/var/folders/wx/1_76tj0s15gc4yxmndvw4l_00000gn/T/RtmpLDdUPF/model-1262f783f7adb.stan', li
##     of arrays by placing brackets after a variable name is deprecated and
```

```
##     will be removed in Stan 2.32.0. Instead use the array keyword before the
##     type. This can be changed automatically using the auto-format flag to
##     stanc
## Warning in '/var/folders/wx/1_76tj0s15gc4yxmndvw4l_00000gn/T/RtmpLDdUPF/model-1262f783f7adb.stan', li
##     of arrays by placing brackets after a variable name is deprecated and
##     will be removed in Stan 2.32.0. Instead use the array keyword before the
##     type. This can be changed automatically using the auto-format flag to
##     stanc

## Running MCMC with 4 sequential chains, with 1 thread(s) per chain...
##
## Chain 1 finished in 0.5 seconds.
## Chain 2 finished in 0.6 seconds.

## Chain 3 Informational Message: The current Metropolis proposal is about to be rejected because of the

## Chain 3 Exception: poisson_lpmf: Rate parameter[4] is nan, but must be nonnegative! (in '/var/folders

## Chain 3 If this warning occurs sporadically, such as for highly constrained variable types like covar

## Chain 3 but if this warning occurs often then your model may be either severely ill-conditioned or mi

## Chain 3

## Chain 3 Informational Message: The current Metropolis proposal is about to be rejected because of the

## Chain 3 Exception: poisson_lpmf: Rate parameter[4] is nan, but must be nonnegative! (in '/var/folders

## Chain 3 If this warning occurs sporadically, such as for highly constrained variable types like covar

## Chain 3 but if this warning occurs often then your model may be either severely ill-conditioned or mi

## Chain 3

## Chain 3 Informational Message: The current Metropolis proposal is about to be rejected because of the

## Chain 3 Exception: poisson_lpmf: Rate parameter[4] is nan, but must be nonnegative! (in '/var/folders

## Chain 3 If this warning occurs sporadically, such as for highly constrained variable types like covar

## Chain 3 but if this warning occurs often then your model may be either severely ill-conditioned or mi

## Chain 3

## Chain 3 Informational Message: The current Metropolis proposal is about to be rejected because of the

## Chain 3 Exception: poisson_lpmf: Rate parameter[4] is nan, but must be nonnegative! (in '/var/folders

## Chain 3 If this warning occurs sporadically, such as for highly constrained variable types like covar

## Chain 3 but if this warning occurs often then your model may be either severely ill-conditioned or mi

## Chain 3

## Chain 3 Informational Message: The current Metropolis proposal is about to be rejected because of the

## Chain 3 Exception: poisson_lpmf: Rate parameter[4] is nan, but must be nonnegative! (in '/var/folders

## Chain 3 If this warning occurs sporadically, such as for highly constrained variable types like covar

## Chain 3 but if this warning occurs often then your model may be either severely ill-conditioned or mi

## Chain 3

## Chain 3 finished in 0.5 seconds.
## Chain 4 finished in 0.4 seconds.
##
```

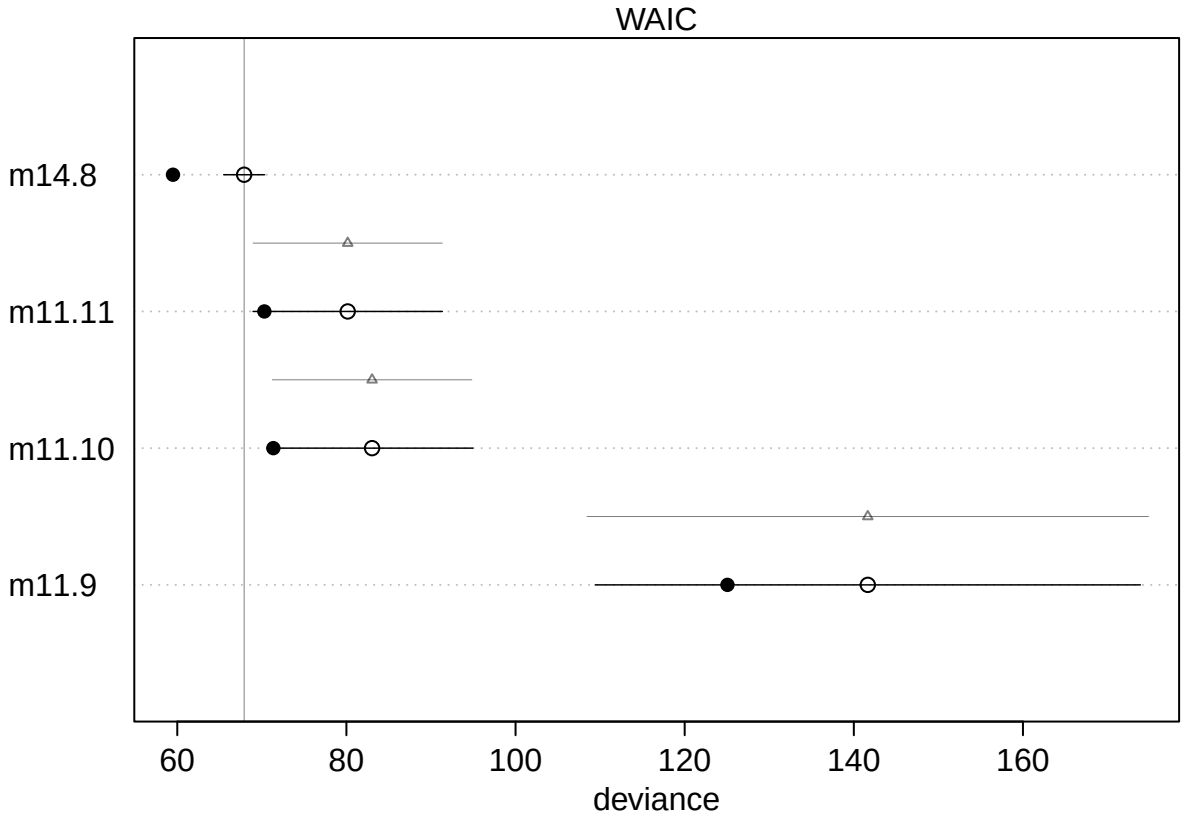
```
## All 4 chains finished successfully.
## Mean chain execution time: 0.5 seconds.
## Total execution time: 2.3 seconds.
```

```
compare( m14.8, m11.9, m11.10, m11.11, func=WAIC )
```

```
##             WAIC        SE    dWAIC      dSE    pWAIC       weight
## m14.8   67.90619  2.419409  0.00000       NA 4.206809 9.972990e-01
## m11.11  80.15385 11.194838 12.24766 11.18341 4.928415 2.184133e-03
## m11.10  83.03604 11.930123 15.12985 11.79404 5.839371 5.169163e-04
## m11.9  141.65118 32.239184 73.74499 33.19204 8.297054 9.667288e-17
```

b) What can you learn about your models through their WAIC scores? In your analysis, pay special attention to the effective number of parameters estimated by WAIC.

```
plot (compare( m14.8, m11.9, m11.10, m11.11, func=WAIC ))
```




```
# The standard error of Gaussian process model is the least among all the models. Also, the WAIC
# score for this model is the least as well. This makes it the best model choice amongst
# the one under consideration.
```

```
# For the result above, we found that the more complex model taking into account spatial
# distances of societies m14.8 outperforms all other models.

# Also the Gaussian process model has less effective parameters (pWAIC) than the simpler model.
# This is a sign of intense regularization on the part of the Gaussian Process model.
```

```
# Taking a look at the effective number of parameters, the order of regularization
# in the priors is as follows:
# m14.8 > m11.11 > m11.10 > m11.9
```