biol792ia4multilevel

morgan hrivnak

2024-12-05

#load packages  
library(rethinking)

## Loading required package: cmdstanr

## This is cmdstanr version 0.8.1.9000

## - CmdStanR documentation and vignettes: mc-stan.org/cmdstanr

## - CmdStan path: C:/Users/morga/.cmdstan/cmdstan-2.35.0

## - CmdStan version: 2.35.0

## Loading required package: posterior

## This is posterior version 1.5.0

##   
## Attaching package: 'posterior'

## The following objects are masked from 'package:stats':  
##   
## mad, sd, var

## The following objects are masked from 'package:base':  
##   
## %in%, match

## Loading required package: parallel

## rethinking (Version 2.42)

##   
## Attaching package: 'rethinking'

## The following object is masked from 'package:stats':  
##   
## rstudent

library(tidyverse)

## Warning: package 'ggplot2' was built under R version 4.4.1

## Warning: package 'tibble' was built under R version 4.4.1

## Warning: package 'stringr' was built under R version 4.4.1

## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──  
## ✔ dplyr 1.1.4 ✔ readr 2.1.5  
## ✔ forcats 1.0.0 ✔ stringr 1.5.1  
## ✔ ggplot2 3.5.1 ✔ tibble 3.2.1  
## ✔ lubridate 1.9.3 ✔ tidyr 1.3.1  
## ✔ purrr 1.0.2

## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()  
## ✖ purrr::map() masks rethinking::map()  
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors

library(raster)

## Loading required package: sp  
##   
## Attaching package: 'raster'  
##   
## The following object is masked from 'package:dplyr':  
##   
## select

library(dplyr)  
library(here)

## Warning: package 'here' was built under R version 4.4.1

## here() starts at C:/Users/morga/OneDrive - The University of South Dakota/Desktop/cupesedimentdata

df <- readRDS(here("alldata\_wrangled.rds"))  
  
#assign non-numerical for percent\_lessorequal  
df$percent\_lessorequal <- as.numeric(df$percent\_lessorequal)

## Warning: NAs introduced by coercion

#filter out NA values  
df\_clean <- df %>%  
 filter(!is.na(percent\_lessorequal))  
#select only the necessary collumns  
df\_cleaner <- df\_clean %>%   
 dplyr::select(percent\_lessorequal, size, site, percent)  
#select CARI and MCDI sites and standardize the x axis  
selected\_sites <- df\_cleaner %>%  
 filter(site %in% c("MCDI", "CARI")) %>%   
 mutate(percent\_standard= standardize(percent))  
print(selected\_sites)

## # A tibble: 24 × 5  
## percent\_lessorequal size site percent percent\_standard  
## <dbl> <dbl> <chr> <dbl> <dbl>  
## 1 6.8 6.8 MCDI 5 -1.29   
## 2 18.1 18.1 MCDI 16 -0.916  
## 3 42.3 42.3 MCDI 50 0.239  
## 4 29.9 29.9 MCDI 50 0.239  
## 5 163. 163. MCDI 84 1.39   
## 6 13 13 MCDI 5 -1.29   
## 7 23.5 23.5 MCDI 16 -0.916  
## 8 53.3 53.3 MCDI 50 0.239  
## 9 4 4 MCDI 16 -0.916  
## 10 28.6 28.6 MCDI 50 0.239  
## # ℹ 14 more rows

#make variable into index  
selected\_sites$cid <-ifelse(selected\_sites$site == "MCDI", 1, 2)

mlmgamma1 <- ulam(  
 alist(  
 size ~ dgamma2(mu, scale) ,  
 log(mu) <- a\_mlm[cid] + b\_mlm[cid] \* percent\_standard ,  
 a\_mlm[cid] ~ dnorm(a, scale\_alpha),  
 b\_mlm[cid] ~ dnorm(b, scale\_beta) ,  
 a ~ dnorm(3.5, 1) ,  
 b ~ dnorm(0.5, 1) ,  
 scale ~ dexp(1) ,  
 scale\_alpha ~ dexp(1) ,  
 scale\_beta ~ dexp(1)  
 ),   
 data = selected\_sites ,  
 iter = 2000,   
 chains = 4 ,  
 warmup = 1000)

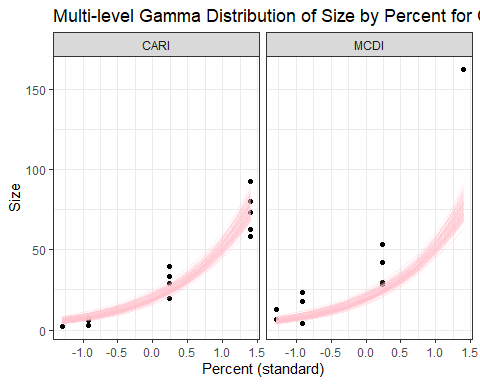
## Running MCMC with 4 sequential chains, with 1 thread(s) per chain...  
##   
## Chain 1 Iteration: 1 / 2000 [ 0%] (Warmup)   
## Chain 1 Iteration: 100 / 2000 [ 5%] (Warmup)   
## Chain 1 Iteration: 200 / 2000 [ 10%] (Warmup)   
## Chain 1 Iteration: 300 / 2000 [ 15%] (Warmup)   
## Chain 1 Iteration: 400 / 2000 [ 20%] (Warmup)   
## Chain 1 Iteration: 500 / 2000 [ 25%] (Warmup)   
## Chain 1 Iteration: 600 / 2000 [ 30%] (Warmup)   
## Chain 1 Iteration: 700 / 2000 [ 35%] (Warmup)   
## Chain 1 Iteration: 800 / 2000 [ 40%] (Warmup)   
## Chain 1 Iteration: 900 / 2000 [ 45%] (Warmup)   
## Chain 1 Iteration: 1000 / 2000 [ 50%] (Warmup)   
## Chain 1 Iteration: 1001 / 2000 [ 50%] (Sampling)   
## Chain 1 Iteration: 1100 / 2000 [ 55%] (Sampling)   
## Chain 1 Iteration: 1200 / 2000 [ 60%] (Sampling)   
## Chain 1 Iteration: 1300 / 2000 [ 65%] (Sampling)   
## Chain 1 Iteration: 1400 / 2000 [ 70%] (Sampling)   
## Chain 1 Iteration: 1500 / 2000 [ 75%] (Sampling)   
## Chain 1 Iteration: 1600 / 2000 [ 80%] (Sampling)   
## Chain 1 Iteration: 1700 / 2000 [ 85%] (Sampling)   
## Chain 1 Iteration: 1800 / 2000 [ 90%] (Sampling)   
## Chain 1 Iteration: 1900 / 2000 [ 95%] (Sampling)   
## Chain 1 Iteration: 2000 / 2000 [100%] (Sampling)   
## Chain 1 finished in 0.8 seconds.  
## Chain 2 Iteration: 1 / 2000 [ 0%] (Warmup)   
## Chain 2 Iteration: 100 / 2000 [ 5%] (Warmup)   
## Chain 2 Iteration: 200 / 2000 [ 10%] (Warmup)   
## Chain 2 Iteration: 300 / 2000 [ 15%] (Warmup)   
## Chain 2 Iteration: 400 / 2000 [ 20%] (Warmup)   
## Chain 2 Iteration: 500 / 2000 [ 25%] (Warmup)   
## Chain 2 Iteration: 600 / 2000 [ 30%] (Warmup)   
## Chain 2 Iteration: 700 / 2000 [ 35%] (Warmup)   
## Chain 2 Iteration: 800 / 2000 [ 40%] (Warmup)   
## Chain 2 Iteration: 900 / 2000 [ 45%] (Warmup)   
## Chain 2 Iteration: 1000 / 2000 [ 50%] (Warmup)   
## Chain 2 Iteration: 1001 / 2000 [ 50%] (Sampling)   
## Chain 2 Iteration: 1100 / 2000 [ 55%] (Sampling)   
## Chain 2 Iteration: 1200 / 2000 [ 60%] (Sampling)   
## Chain 2 Iteration: 1300 / 2000 [ 65%] (Sampling)   
## Chain 2 Iteration: 1400 / 2000 [ 70%] (Sampling)   
## Chain 2 Iteration: 1500 / 2000 [ 75%] (Sampling)   
## Chain 2 Iteration: 1600 / 2000 [ 80%] (Sampling)   
## Chain 2 Iteration: 1700 / 2000 [ 85%] (Sampling)   
## Chain 2 Iteration: 1800 / 2000 [ 90%] (Sampling)   
## Chain 2 Iteration: 1900 / 2000 [ 95%] (Sampling)   
## Chain 2 Iteration: 2000 / 2000 [100%] (Sampling)   
## Chain 2 finished in 0.6 seconds.  
## Chain 3 Iteration: 1 / 2000 [ 0%] (Warmup)   
## Chain 3 Iteration: 100 / 2000 [ 5%] (Warmup)   
## Chain 3 Iteration: 200 / 2000 [ 10%] (Warmup)   
## Chain 3 Iteration: 300 / 2000 [ 15%] (Warmup)   
## Chain 3 Iteration: 400 / 2000 [ 20%] (Warmup)   
## Chain 3 Iteration: 500 / 2000 [ 25%] (Warmup)   
## Chain 3 Iteration: 600 / 2000 [ 30%] (Warmup)   
## Chain 3 Iteration: 700 / 2000 [ 35%] (Warmup)   
## Chain 3 Iteration: 800 / 2000 [ 40%] (Warmup)   
## Chain 3 Iteration: 900 / 2000 [ 45%] (Warmup)   
## Chain 3 Iteration: 1000 / 2000 [ 50%] (Warmup)   
## Chain 3 Iteration: 1001 / 2000 [ 50%] (Sampling)   
## Chain 3 Iteration: 1100 / 2000 [ 55%] (Sampling)   
## Chain 3 Iteration: 1200 / 2000 [ 60%] (Sampling)   
## Chain 3 Iteration: 1300 / 2000 [ 65%] (Sampling)   
## Chain 3 Iteration: 1400 / 2000 [ 70%] (Sampling)   
## Chain 3 Iteration: 1500 / 2000 [ 75%] (Sampling)   
## Chain 3 Iteration: 1600 / 2000 [ 80%] (Sampling)   
## Chain 3 Iteration: 1700 / 2000 [ 85%] (Sampling)   
## Chain 3 Iteration: 1800 / 2000 [ 90%] (Sampling)   
## Chain 3 Iteration: 1900 / 2000 [ 95%] (Sampling)   
## Chain 3 Iteration: 2000 / 2000 [100%] (Sampling)   
## Chain 3 finished in 1.7 seconds.  
## Chain 4 Iteration: 1 / 2000 [ 0%] (Warmup)   
## Chain 4 Iteration: 100 / 2000 [ 5%] (Warmup)   
## Chain 4 Iteration: 200 / 2000 [ 10%] (Warmup)   
## Chain 4 Iteration: 300 / 2000 [ 15%] (Warmup)   
## Chain 4 Iteration: 400 / 2000 [ 20%] (Warmup)   
## Chain 4 Iteration: 500 / 2000 [ 25%] (Warmup)   
## Chain 4 Iteration: 600 / 2000 [ 30%] (Warmup)   
## Chain 4 Iteration: 700 / 2000 [ 35%] (Warmup)   
## Chain 4 Iteration: 800 / 2000 [ 40%] (Warmup)   
## Chain 4 Iteration: 900 / 2000 [ 45%] (Warmup)   
## Chain 4 Iteration: 1000 / 2000 [ 50%] (Warmup)   
## Chain 4 Iteration: 1001 / 2000 [ 50%] (Sampling)   
## Chain 4 Iteration: 1100 / 2000 [ 55%] (Sampling)   
## Chain 4 Iteration: 1200 / 2000 [ 60%] (Sampling)   
## Chain 4 Iteration: 1300 / 2000 [ 65%] (Sampling)   
## Chain 4 Iteration: 1400 / 2000 [ 70%] (Sampling)   
## Chain 4 Iteration: 1500 / 2000 [ 75%] (Sampling)   
## Chain 4 Iteration: 1600 / 2000 [ 80%] (Sampling)   
## Chain 4 Iteration: 1700 / 2000 [ 85%] (Sampling)   
## Chain 4 Iteration: 1800 / 2000 [ 90%] (Sampling)   
## Chain 4 Iteration: 1900 / 2000 [ 95%] (Sampling)   
## Chain 4 Iteration: 2000 / 2000 [100%] (Sampling)   
## Chain 4 finished in 0.7 seconds.  
##   
## All 4 chains finished successfully.  
## Mean chain execution time: 1.0 seconds.  
## Total execution time: 4.6 seconds.

precis(mlmgamma1, depth= 2)

## mean sd 5.5% 94.5% rhat ess\_bulk  
## a\_mlm[1] 3.4826009 0.10427679 3.31691670 3.644179 1.002077 1848.2241  
## a\_mlm[2] 2.9736425 0.11926286 2.78705940 3.165219 1.004445 1541.5055  
## b\_mlm[1] 1.0789963 0.10346693 0.91766613 1.242051 1.002862 1934.3983  
## b\_mlm[2] 0.9795103 0.09085063 0.83535876 1.129866 1.004114 1495.0287  
## a 3.2611732 0.42487571 2.62469610 3.931721 1.002178 1530.0985  
## b 0.9806793 0.29884189 0.51021300 1.362968 1.002830 1439.8306  
## scale 2.9887341 0.80976882 1.88145510 4.396731 1.001919 2033.8576  
## scale\_alpha 0.6721660 0.52529646 0.17391752 1.675836 1.002725 1026.2661  
## scale\_beta 0.3717991 0.42219906 0.03643116 1.121305 1.005161 749.3177

posts = extract.samples(mlmgamma1)  
  
posts\_table = tibble(a1 = posts$a\_mlm[,1],  
 a2 = posts$a\_mlm[,2],  
 b1 = posts$b\_mlm[,1],  
 b2 = posts$b\_mlm[,2]) %>%   
 mutate(iter = 1:nrow(.))  
  
post\_preds = posts\_table %>%   
 filter(iter <= 100) %>%  
 expand\_grid(percent\_standard = seq(min(selected\_sites$percent\_standard),   
 max(selected\_sites$percent\_standard),  
 length.out = 20)) %>%   
 mutate(size = exp(a2 + b2\*percent\_standard))

ggplot(data =selected\_sites %>% filter(site %in% c("CARI", "MCDI")), aes(x = percent\_standard, y = size)) +  
 geom\_point() +  
 theme\_bw() +  
 geom\_line(data = post\_preds, aes(group = iter), color="pink", alpha = 0.2) + #this needs to change for the data  
facet\_wrap(~site) + # Create separate panels for each site  
 labs(title = "Multi-level Gamma Distribution of Size by Percent for CARI and MCDI",  
 x = "Percent (standard)",  
 y = "Size")



MCDI is a NEON site in Flint Hills region of Kansas south of Manhattan, KS. CARI is a NEON site in central Alaska at the CPCRW northeast of Fairbanks, Alaska. This examines the interaction of the size of substrate present in total sample percent between sites MCDI and CARI. The data is not standardized and was put into a gamma distribution. The x-axis percent was standardized to better examine the trend in data. The data falls into a gamma distribution because it is positive and continuous. A multi-level model was used to increase the sample size without increasing the number of observations as the data was pulled from NEON data base, it allows us to further improve the model compared to a regular MCMC chain.

mlmgamma1 uses a multi-level gamma distribution to examine the trends of CARI and MCDI. The slope has a mu of 3.5 and a scale of 1 because these are the exponent of that best fits the data set. The multi-level gamma is logged so the priors need to reflect that relationship in their calculation. The intercept has a mu of 0.5 and a scale of 1 for a similar reason to the MCDI has a mean intercept of 3.49 (CL89% 3.31-3.66) and a mean slope of 1.08 (CL89% 0.92-1.26). CARI has a mean intercept of 2.97 (CL89% 2.78-3.16) and a mean slope of 0.98 (CL89% 0.83-1.13).

This relationship makes biological sense as it should be positive. It should be a positive relationship as you go along the x-axis it encompasses more of the samples for that given site. The rate in which it increases tells us that MCDI has more larger substrate (cobbles) than CARI. CARI has more finer substrate than MCDI. Geologically speaking this also makes sense as CARI is located in Alaska while MCDI is located in Kansas. These streams have very different geological process affecting them as well as human activity, MAT, and MAP. CARI. We would expect this kind of substrate at CARI to have glacial till and for soil to be affected by permafrost thaws. This is hard to examine looking at NEON images of this site in particular due to the water level. At MCDI you can see that generally where they survey the stream it has high banks on one side and what appears to be gravel or small cobbles on the other bank. MCDI is characterized as a metavolcanic bedrock while CARI is characterized as Birtch Creek Schist. Although both of them are composed of silt CARI has soils that are typically silty alluvium and are subject to permafrost layers while MCDI has Ivan silt loam that is prone to occasional flooding. MCDI is typical upland headwaters stream while CARI is a second-order perennial stream. The two streams are not only geologically and hydrologically different from one another but their soils and regular disturbance events are drastically different from one another.