Basic CAR Model

Alvin Sheng

6/30/2021

```
library(here)
```

```
## here() starts at /Users/Alvin/Documents/NCSU_Fall_2021/NIH_SIP/flood-risk-health-effects
```

```
library(coda)
library(CARBayes)
```

```
## Loading required package: MASS

## Loading required package: Rcpp

## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

CAR model results

Inference is based on 3 markov chains, each of which has been run for 100000 samples, the first 10000 of which has been removed for burn-in. The remaining 90000 samples are thinned by 5, resulting in 18000 * 3 = 54000 samples for inference across the 3 Markov chains.

```
load(here("modeling_files/model_3chains_var_exclude.RData"))
```

Output for the first chain is shown below.

```
chain1
```

```
##
## #################
## #### Model fitted
## #################
## Likelihood model - Gaussian (identity link function)
## Random effects model - Leroux CAR
## Regression equation - Y ~ X
## <environment: 0x7fd4b8b61700>
## Number of missing observations - 0
##
## ############
## #### Results
## ############
## Posterior quantities and DIC
##
##                          Median    2.5%   97.5% n.effective Geweke.diag
## (Intercept)             77.7453 77.7253 77.7651     18000.0         1.9
```

```
## Xpct_fs_risk_2020_5     -0.1179 -0.2087 -0.0283     10734.0         1.1
## Xpct_floodfactor2        0.0171 -0.0283  0.0625     12960.6         0.8
## Xpct_floodfactor3       -0.0154 -0.0617  0.0308     12684.5        -0.5
## Xpct_floodfactor4        0.0479 -0.0024  0.0992      9935.2        -0.4
## Xpct_floodfactor5       -0.0196 -0.0827  0.0441     11001.7        -1.0
## Xpct_floodfactor6        0.0262 -0.0392  0.0918     12597.8         1.1
## Xpct_floodfactor7       -0.0120 -0.0623  0.0395     10856.9        -0.9
## Xpct_floodfactor8       -0.0144 -0.0583  0.0289     14894.6         0.1
## Xpct_floodfactor9        0.0877  0.0139  0.1609     11986.6        -1.1
## Xavg_risk_fsf_2020_100   0.1202  0.0388  0.2000      8785.6         0.2
## Xavg_risk_score_sfha     0.0083 -0.0430  0.0588     12590.9         1.9
## Xavg_risk_score_no_sfha -0.0063 -0.0954  0.0819     10668.5         0.3
## XEP_POV                 -0.1779 -0.2448 -0.1106     11811.2         0.4
## XEP_UNEMP               -0.0596 -0.1079 -0.0106     12352.9         1.1
## XEP_PCI                  0.2465  0.1733  0.3203     10025.0         0.5
## XEP_NOHSDP              -0.0541 -0.1368  0.0286      8373.5         0.0
## XEP_DISABL              -0.1211 -0.1789 -0.0630      9023.2        -1.2
## XEP_SNGPNT              -0.2182 -0.2656 -0.1727     14142.0         2.8
## XEP_MINRTY              -0.3378 -0.4233 -0.2514      5446.1        -0.7
## XEP_LIMENG               0.3492  0.2820  0.4150      8736.9        -0.3
## XEP_MUNIT                0.1100  0.0494  0.1697      8593.8        -0.1
## XEP_MOBILE              -0.0883 -0.1493 -0.0271      7809.0         0.8
## XEP_CROWD               -0.0735 -0.1215 -0.0249      9235.4        -1.1
## XEP_NOVEH               -0.1260 -0.1793 -0.0722     10100.2        -0.1
## XEP_GROUPQ               0.1105  0.0758  0.1454     14218.3         0.7
## XEP_UNINSUR             -0.0314 -0.0895  0.0286      7662.7        -0.4
## Xco                     -0.0954 -0.1659 -0.0265      4192.4         1.0
## Xno2                    -0.0193 -0.1204  0.0829      3562.1        -0.7
## Xo3                     -0.0394 -0.1733  0.0929       943.6        -0.3
## Xpm10                    0.1123  0.0308  0.1929      3248.1        -0.5
## Xpm25                   -0.2881 -0.4125 -0.1668      1405.1         0.6
## Xso2                    -0.0758 -0.1279 -0.0248      6256.1         1.3
## Xtotal_mean             -0.9107 -0.9826 -0.8377      6328.3         0.3
## nu2                      0.3197  0.2576  0.3815      1883.7        -0.3
## tau2                     1.8848  1.5878  2.2258      1910.0         0.1
## rho                      0.9924  0.9753  0.9992      7881.1         0.8
##
## DIC =  6926.558       p.d =  1660.539       LMPL =  -3825.94
```

The smallest effective sample size is 935.8, for ozone (o3).

```
chain1$accept
```

```
##     beta      phi      nu2     tau2      rho
## 100.0000 100.0000 100.0000 100.0000  45.2697
```

It appears that beta, phi, nu2, and tau2 probably have Gibbs steps, whereas rho has a Metropolis-Hastings step. In any case, the acceptance probabilities are acceptable.

Model Diagnostics

Beta samples

```r
beta_samples <- mcmc.list(chain1$samples$beta, chain2$samples$beta,
                          chain3$samples$beta)
```

```r
saveRDS(beta_samples, file = here("modeling_files/model_3chains_var_exclude_beta_samples.rds"))
```

```r
plot(beta_samples)
```

```r
gelman.diag(beta_samples)
```

```
## Potential scale reduction factors:
##
##       Point est. Upper C.I.
##  [1,]          1          1
##  [2,]          1          1
##  [3,]          1          1
##  [4,]          1          1
##  [5,]          1          1
##  [6,]          1          1
##  [7,]          1          1
##  [8,]          1          1
##  [9,]          1          1
## [10,]          1          1
## [11,]          1          1
## [12,]          1          1
## [13,]          1          1
## [14,]          1          1
## [15,]          1          1
## [16,]          1          1
## [17,]          1          1
## [18,]          1          1
## [19,]          1          1
## [20,]          1          1
## [21,]          1          1
## [22,]          1          1
## [23,]          1          1
## [24,]          1          1
## [25,]          1          1
## [26,]          1          1
## [27,]          1          1
## [28,]          1          1
## [29,]          1          1
## [30,]          1          1
## [31,]          1          1
## [32,]          1          1
## [33,]          1          1
## [34,]          1          1
##
## Multivariate psrf
##
## 1
```

Examining tau2, nu2, rho

```
tau2_samples <- mcmc.list(chain1$samples$tau2, chain2$samples$tau2,
                          chain3$samples$tau2)

nu2_samples <- mcmc.list(chain1$samples$nu2, chain2$samples$nu2,
                          chain3$samples$nu2)

rho_samples <- mcmc.list(chain1$samples$rho, chain2$samples$rho,
                          chain3$samples$rho)
```

```
plot(tau2_samples)

plot(nu2_samples)

plot(rho_samples)
```

```
gelman.diag(tau2_samples)
```

```
## Potential scale reduction factors:
##
##      Point est. Upper C.I.
## [1,]          1          1
```

```
gelman.diag(nu2_samples)
```

```
## Potential scale reduction factors:
##
##      Point est. Upper C.I.
## [1,]          1          1
```

```
gelman.diag(rho_samples)
```

```
## Potential scale reduction factors:
##
##      Point est. Upper C.I.
## [1,]          1          1
```

Examining a sample of the 3108 phi parameters

```
phi_samples <- mcmc.list(chain1$samples$phi, chain2$samples$phi, chain3$samples$phi)
```

```
set.seed(1157, kind = "Mersenne-Twister", normal.kind = "Inversion", sample.kind = "Rejection")

phi_subset_idx <- sample(1:3108, size = 10)

phi_samples_subset <- phi_samples[, phi_subset_idx]
```

```
plot(phi_samples_subset)
```

```
gelman.diag(phi_samples_subset)
```

```
## Potential scale reduction factors:
##
##       Point est. Upper C.I.
```

```
##  [1,]          1          1
##  [2,]          1          1
##  [3,]          1          1
##  [4,]          1          1
##  [5,]          1          1
##  [6,]          1          1
##  [7,]          1          1
##  [8,]          1          1
##  [9,]          1          1
## [10,]          1          1
##
## Multivariate psrf
##
## 1
```

Inference

```r
beta_samples_matrix <- rbind(chain1$samples$beta, chain2$samples$beta, chain3$samples$beta)

colnames(beta_samples_matrix) <- colnames(chain1$X)
```

```r
(beta_inference <- round(t(apply(beta_samples_matrix, 2, quantile, c(0.5, 0.025, 0.975))),5))
```

```
##                             50%     2.5%    97.5%
## (Intercept)            77.74541 77.72526 77.76547
## Xpct_fs_risk_2020_5    -0.11846 -0.20947 -0.02771
## Xpct_floodfactor2       0.01733 -0.02848  0.06292
## Xpct_floodfactor3      -0.01574 -0.06166  0.02993
## Xpct_floodfactor4       0.04814 -0.00219  0.09916
## Xpct_floodfactor5      -0.02001 -0.08280  0.04319
## Xpct_floodfactor6       0.02603 -0.03848  0.09156
## Xpct_floodfactor7      -0.01147 -0.06218  0.03928
## Xpct_floodfactor8      -0.01455 -0.05858  0.02901
## Xpct_floodfactor9       0.08748  0.01451  0.16079
## Xavg_risk_fsf_2020_100  0.12019  0.03839  0.20060
## Xavg_risk_score_sfha    0.00802 -0.04303  0.05908
## Xavg_risk_score_no_sfha -0.00614 -0.09451  0.08119
## XEP_POV                -0.17780 -0.24495 -0.11046
## XEP_UNEMP              -0.05955 -0.10768 -0.01091
## XEP_PCI                 0.24681  0.17371  0.32026
## XEP_NOHSDP             -0.05413 -0.13624  0.02840
## XEP_DISABL             -0.12113 -0.17901 -0.06317
## XEP_SNGPNT             -0.21845 -0.26570 -0.17201
## XEP_MINRTY             -0.33722 -0.42218 -0.25146
## XEP_LIMENG              0.34887  0.28217  0.41493
## XEP_MUNIT               0.11002  0.04950  0.17033
## XEP_MOBILE             -0.08845 -0.14917 -0.02743
## XEP_CROWD              -0.07362 -0.12170 -0.02504
## XEP_NOVEH              -0.12598 -0.17966 -0.07259
## XEP_GROUPQ              0.11039  0.07611  0.14525
## XEP_UNINSUR            -0.03139 -0.09039  0.02834
## Xco                    -0.09513 -0.16489 -0.02588
## Xno2                   -0.01956 -0.12031  0.08103
```

```
## Xo3                        -0.04186 -0.17364  0.09052
## Xpm10                       0.11263  0.03151  0.19444
## Xpm25                      -0.28796 -0.41248 -0.16528
## Xso2                       -0.07616 -0.12793 -0.02453
## Xtotal_mean                -0.91055 -0.98245 -0.83768
```

List of significant beta coefficients:

```
colnames(beta_samples_matrix)[sign(beta_inference[, 2]) == sign(beta_inference[, 3])]
```

```
##  [1] "(Intercept)"             "Xpct_fs_risk_2020_5"    "Xpct_floodfactor9"
##  [4] "Xavg_risk_fsf_2020_100" "XEP_POV"                "XEP_UNEMP"
##  [7] "XEP_PCI"                 "XEP_DISABL"             "XEP_SNGPNT"
## [10] "XEP_MINRTY"              "XEP_LIMENG"             "XEP_MUNIT"
## [13] "XEP_MOBILE"              "XEP_CROWD"              "XEP_NOVEH"
## [16] "XEP_GROUPQ"              "Xco"                    "Xpm10"
## [19] "Xpm25"                   "Xso2"                   "Xtotal_mean"
```

My sparse implementation

```
load(here("modeling_files/model_1chain_var_exclude_sparse.RData"))
```

```
chain1$modelfit
```

```
##           DIC          p.d         WAIC          p.w         LMPL
##      6913.325     1649.640     6876.002     1227.458    -3806.863
## loglikelihood
##     -1807.022
```

```
mcmc_samps <- chain1$samples

effectiveSize(mcmc_samps$beta)
```

```
##       var1       var2       var3       var4       var5       var6       var7
## 17824.8740 10588.2173 13177.1579 13446.7282 10459.9458 10412.9187 11606.2789
##       var8       var9      var10      var11      var12      var13      var14
## 11633.0534 14869.8877 11052.7277  9542.8497 11781.1272 10761.8732 12304.7678
##      var15      var16      var17      var18      var19      var20      var21
## 12082.5510  9529.2160  9427.1320  9654.0071 13385.0412  4968.8319 10662.4156
##      var22      var23      var24      var25      var26      var27      var28
##  9006.2581  8618.2333 10854.5943 10996.9025 14468.5769  8008.8227  4197.2541
##      var29      var30      var31      var32      var33      var34
##  3307.6520   873.2518  3393.8121  1465.3655  6072.1736  6791.8248
```

It's easier to achieve a high sample size. I can have 10x fewer iterations.

```
effectiveSize(mcmc_samps$sigma2)
```

```
##     var1
## 1982.105
```

```
effectiveSize(mcmc_samps$nu2)
```

```
##   var1
## 2003.7
```

```r
effectiveSize(mcmc_samps$rho)
```

```
##     var1
## 8737.944
```

```r
effectiveSize(mcmc_samps$Y)
```

```
##     var1     var2     var3     var4     var5     var6     var7     var8
## 14958.76 16219.32 16425.60 15512.36 15399.18 15819.92 15661.60 16051.90
##     var9    var10
## 16502.47 16298.51
```

```r
t(apply(mcmc_samps$beta, 2, quantile, c(0.5, 0.025, 0.975)))
```

```
##                50%         2.5%       97.5%
## var1  77.746506148 77.726411457 77.76645330
## var2  -0.119540299 -0.210452554 -0.02909804
## var3   0.017900183 -0.027865030  0.06325522
## var4  -0.016757355 -0.062849320  0.02888706
## var5   0.050129231 -0.001709434  0.10149961
## var6  -0.020836773 -0.084061452  0.04178001
## var7   0.027238928 -0.037526315  0.09284415
## var8  -0.010973621 -0.060695936  0.04078919
## var9  -0.015118646 -0.059089625  0.02768246
## var10  0.086282980  0.013777734  0.16009979
## var11  0.121360779  0.042087761  0.20293893
## var12  0.008158981 -0.043258548  0.05890849
## var13 -0.006309570 -0.095211432  0.08202875
## var14 -0.178166872 -0.244198785 -0.11073784
## var15 -0.061460637 -0.110337992 -0.01330997
## var16  0.244703557  0.170855815  0.31917766
## var17 -0.053992619 -0.137481557  0.02888784
## var18 -0.118966450 -0.177010691 -0.06076987
## var19 -0.217929506 -0.264566749 -0.17075672
## var20 -0.337053185 -0.422677017 -0.25329809
## var21  0.347558929  0.282064515  0.41340951
## var22  0.111408456  0.050665553  0.17062336
## var23 -0.087001928 -0.146387155 -0.02668316
## var24 -0.074577134 -0.123766678 -0.02567313
## var25 -0.127042726 -0.180567305 -0.07353508
## var26  0.109783964  0.074951764  0.14426383
## var27 -0.028954755 -0.088080431  0.03103675
## var28 -0.093294506 -0.163674562 -0.02383003
## var29 -0.021817409 -0.119213178  0.07528710
## var30 -0.050301272 -0.186201008  0.08593165
## var31  0.115483751  0.034014079  0.19704008
## var32 -0.287547411 -0.407863043 -0.16612464
## var33 -0.078473225 -0.130344934 -0.02708925
## var34 -0.913616225 -0.986850856 -0.84071714
```

```r
quantile(mcmc_samps$nu2, c(0.5, 0.025, 0.975))
```

```
##       50%      2.5%     97.5%
## 0.3216937 0.2609731 0.3824641
```

```r
quantile(mcmc_samps$sigma2, c(0.5, 0.025, 0.975))
```

```
##      50%     2.5%    97.5%
## 1.875368 1.576425 2.213394
```

```r
quantile(mcmc_samps$rho, c(0.5, 0.025, 0.975))
```

```
##       50%      2.5%     97.5%
## 0.9927457 0.9762306 0.9992126
```

Imputed Y values

```r
t(apply(mcmc_samps$Y, 2, quantile, c(0.5, 0.025, 0.975)))
```

```
##            50%     2.5%    97.5%
## var1  79.53963 77.91262 81.15431
## var2  77.62478 75.98932 79.26445
## var3  75.73865 74.12137 77.34398
## var4  80.92168 79.21596 82.62262
## var5  82.47714 80.79385 84.17298
## var6  76.85689 75.17219 78.55423
## var7  75.78624 74.16714 77.41043
## var8  81.49117 79.95209 83.05497
## var9  77.56431 75.99440 79.11463
## var10 76.68195 75.07241 78.32694
```