```r
library(npreg)

## Package 'npreg' version 1.1.0
## Type 'citation("npreg")' to cite this package.

suppressMessages(library(tidyverse))

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

library(rstan)

## Loading required package:  StanHeaders
##
## rstan version 2.32.6 (Stan version 2.32.2)
## 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)
## Do not specify '-march=native' in 'LOCAL_CPPFLAGS' or a Makevars
file
##
## Attaching package:  'rstan'
## The following object is masked from 'package:tidyr':
##
##     extract

library(Pareto)
library(MESS)
library(scales)

##
## Attaching package:  'scales'
## The following object is masked from 'package:purrr':
##
##     discard
## The following object is masked from 'package:readr':
##
##     col_factor

library(readxl)
library(gridExtra)
```

```
##
## Attaching package:  'gridExtra'
## The following object is masked from 'package:dplyr':
##
##     combine
```

```r
library(colorspace)
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```stan
data {
    int <lower=1> n; // Sample size
    int <lower=1> p; // regression coefficient size
    int <lower=1> psi; // splines coefficient size
    real <lower=0> u; // large threshold value
    matrix[n,p] bsLinear; // fwi dataset
    matrix[n, (psi*p)] bsNonlinear; // thin plate splines basis
    matrix[n,p] xholderLinear; // fwi dataset
    matrix[n, (psi*p)] xholderNonlinear; // thin plate splines basis
    array[n] real <lower=1> y; // extreme response
    real <lower=0> atau;
    matrix[2, (2*p)] basisFL;
    array[(p*2)] int indexFL;
}
parameters {
    vector[(p+1)] theta; // linear predictor
    array[p] vector[(psi-2)] gammaTemp; // constraint splines coefficient from 2 to psi-1
    real <lower=0> lambda1; // lasso penalty
    real <lower=0> lambda2; // group lasso penalty
    array[p] real <lower=0> tau;
}

transformed parameters {
    array[n] real <lower=0> alpha; // covariate-adjusted tail index
    array[n] real <lower=0> newalpha; // new tail index
    array[p] vector[psi] gamma; // splines coefficient
    matrix[n, p] newgnl; // nonlinear component
    matrix[n, p] newgl; // linear component
    matrix[n, p] newgsmooth; // linear component

    {
      array[p] vector[2] gammaFL;
      matrix[2, p] subgnl;
      matrix[n, p] gnl; // nonlinear component
      matrix[n, p] gl; // linear component
      matrix[n, p] gsmooth; // linear component


      for(j in 1:p){
          gamma[j][2:(psi-1)] = gammaTemp[j][1:(psi-2)];
          subgnl[,j] = bsNonlinear[indexFL[(((j-1)*2)+1):(((j-1)*2)+2)], (((j-1)*psi)+2):(((
          gammaFL[j] = basisFL[, (((j-1)*2)+1):(((j-1)*2)+2)] * subgnl[,j] * (-1);
          gamma[j][1] = gammaFL[j][1];
          gamma[j][psi] = gammaFL[j][2];
      };

      for (j in 1:p){
          gnl[,j] = bsNonlinear[,(((j-1)*psi)+1):(((j-1)*psi)+psi)] * gamma[j];
          newgnl[,j] = xholderNonlinear[,(((j-1)*psi)+1):(((j-1)*psi)+psi)] * gamma[j];
          gl[,j] = bsLinear[,j] * theta[j+1];
          newgl[,j] = xholderLinear[,j] * theta[j+1];
          gsmooth[,j] = gl[,j] + gnl[,j];
          newgsmooth[,j] = newgl[,j] + newgnl[,j];
      };

      for (i in 1:n){
          alpha[i] = exp(theta[1] + sum(gsmooth[i,]));
          newalpha[i] = exp(theta[1] + sum(newgsmooth[i,]));
      };
    }
}

model {
    // likelihood
    for (i in 1:n){
        target += pareto_lpdf(y[i] | u, alpha[i]);
    }
    target += normal_lpdf(theta[1] | 0, 100);
    target += gamma_lpdf(lambda1 | 1, 1e-3);
    target += gamma_lpdf(lambda2 | 1, 1e-3);
    target += (2*p*log(lambda2));
    for (j in 1:p){
        target += double_exponential_lpdf(theta[(j+1)] | 0, lambda1);
        target += gamma_lpdf(tau[j] | atau, lambda2^2*0.5);
        target += multi_normal_lpdf(gamma[j] | rep_vector(0, psi), diag_matrix(rep_vector(1,
}
"

## Error:  <text>:1:6:  unexpected '{'
## 1:  data {
```

```
##           ^
```