# Emulating computer experiments of transport infrastructure slope stability using Gaussian processes and Bayesian inference

Corresponding author

22/03/2021

## Example plots

This document includes instructions to reproduce Figures 1 and S4. The code is written in R language.

### Preamble

Firstly, set the working directory to store the datasets and R scripts. A few data transformations are also performed:

- Slope angles in degrees are transformed angle cotangent
- Permeability is scaled by 1E8
- Time to failure (TTF) is transformed using a square root

To avoid code evaluation, set `eval=FALSE` in the chunks that you wish to avoid.

```
empty <- function(x,y){
  plot(x,y,xlab="",ylab = "", main="",type="n",axes = FALSE)
}
AN <- function(x) {return(as.numeric(x))}

#Inputs to the emulator
INPUTS=read.csv("INPUTS.csv",stringsAsFactors=TRUE,row.names=1)

#Time to failure (TTF)/output
TTF=read.csv("TTF.csv",stringsAsFactors=TRUE,row.names=1)

#Data matrix
DATA=data.frame(height=INPUTS$Height,angle=1/tan(INPUTS$Angle*pi/180),
                peakc=INPUTS$Cohesion,peakf=INPUTS$Friction,
                perm=1E8*INPUTS$Perm,ttf=TTF)

#The experiments whose TTF is greater that 183.997 is deemed as censored.
DATA$TTF[DATA$TTF>183.997]<-NA
DATA$TTF <- DATA$TTF^(1/2)

#Define some indicator variables so that the censored and uncensored
#observations are easily distinguishable
out<-c(); for(i in 1:nrow(DATA)) ifelse(is.na(DATA$TTF[i]),out[i]<-0,out[i]<-1)
DATA$status<-out
DATA$iscensored<-1-DATA$status
DATA$cen<-DATA$TTF+DATA$status
```

```
testDATA <-DATA[DATA$status==1,]
testDATAC <-DATA[DATA$status==0,]
```

### Performing the MCMC inference

If you want to produce your own Markov chain Monte Carlo results, run the following chunk of code. Alternatively, skip this section and upload the RDS file "FULLsqM5N.rds", its specification will be given in the next section.

It is recommended that you first run the chunk below at a low number of iterations iter and adaptive (a few thousand). For example, use iter=3e3 and adaptive=1e3, which will give a total of 3e3 iterations, of which 1e3 are adaptive. In our research, we used iter=1e5+2e5 and adaptive=1e5.

```
packages <- c("invgamma", "mvtnorm", "plgp",
              "TruncatedNormal", "TruncatedNormal",
              "condMVNorm", "Rfast","invgamma")
install.packages(setdiff(packages, rownames(installed.packages())),
                 repos="https://www.stats.bris.ac.uk/R/")

library("invgamma")
library("mvtnorm")
library("plgp")
library("TruncatedNormal")
library("condMVNorm")
library("Rfast")
library("invgamma")

#MCMCMetFullNugget includes the emulator MCMC algorithm called GPEMetFull
source("MCMCMetFullNugget.R")

#MCMCfunctions includes the functions required by the emulator
source("MCMCfunctions.r")#MCMC functions, prior distributions, etc

#Start.time and End.time are for reference only
#desc includes a copy of the arguments

#The below arguments are equivalent to those used to produce the FULLsqM5N.rds file,
#except the number of iterations (including adaptive) is lower.

##The below set-up will take a few minutes to run.
##WARNING!! 3e5 iterations will take about 12 hours to run on a machine with 8 cores and 32GB RAM.
##Execution time mainly depends on the number of iterations
##WARNING! The sampler currently works only with q=0 (zero mean), q=1 (constant mean) and q=6
##(full regressor mean).

##Define the burn-in/ADAPTIVE size and total iteration size

ADAPTIVE <- 1e3
ITER <- ADAPTIVE + 2e3

Start.time<-paste(c("Start at ",format(Sys.time(), "%d-%b-%Y %H:%M")),sep="",collapse="")
FULLsqM5Ntest<-GPEMetFull(y=testDATA[,6],X=testDATA[,1:5],C=testDATAC[,1:5],
                      BETApriorM=rep(0,6),BETApriorV=c(100,rep(16,5)),
```

```
                        BETAv=rep(1,6),S2priorM=11.11111,S2priorV=15.4321,S2v=1,
                        CLprior=rep(0.2,5),CLv=rep(0.1,5),CEN=(183.997)^(1/2),
                        Nv=0.1,iter=ITER,adaptive=ADAPTIVE,q=6,
                        CORRFUN = "Matern5_2",ZEROMEAN=FALSE)
```

```
##   |                                                                      |
```

```
End.time<-paste(c("End at ",format(Sys.time(), "%d-%b-%Y %H:%M")),sep="",collapse="")
desc<-"sq rt ttf, IGa sig prior. y=testDATA[,6],X=testDATA[,1:5],C=testDATAC[,1:5],
                        BETApriorM=rep(0,6),BETApriorV=c(100,rep(16,5)),
                        BETAv=rep(1,6),S2priorM=11.11111,S2priorV=15.4321,S2v=1,
                        CLprior=rep(0.2,5),CLv=rep(0.1,5),CEN=(183.997)^(1/2),
                        Nv=0.1,iter=ITER,adaptive=ADAPTIVE,q=6,
                        CORRFUN = Matern5_2,ZEROMEAN=FALSE"
FULLsqM5Ntest[[6]] <- paste(c(Start.time,desc,End.time),sep=" ",collapse=" ")
#saveRDS(FULLsqM5Ntest,"FULLsqM5Ntest")
```

### Arguments used for creating the "FULLsqM5N.rds" file

This file is equivalent to the one used in the results and discussion sections of the manuscript

- y- observed output
- X- input matrix corresponding to the observed output
- C- input matrix corresponding to the censored output
- BETApriorM, BETApriorV and BETAv are the prior means, prior variances and tuning variances of the regressor coefficients $\beta$
- S2priorM, S2priorV and S2v are the prior mean, prior variance and tuning variance of the marginal variance $\sigma^2$. The chosen prior mean and variance correspond to the parameters of the $\sigma^2$ prior distribution, which is Inverse-Gamma(10,100).
- CLprior and CLv are the correlation length $\theta$ prior parameters and tuning variances
- CEN is the censoreing limit
- Nv is the tuning variance of the nugget $\tau$, its prior distribution is Inverse-Gamma(3,1) and is defined locally in GPEMetFull.
- iter- the number of iterations (including the adaptibe/burn-in period)
- adaptive- the number of adaptive iterations
- q- the dimension of the regressor function. Currently, the sampler works only with q=0 (zero mean), q=1 (constant mean) and q=6 (full regressor mean).
- CORRFUN- correlation function type. Can be one of "Gaussian", "Matern3_2" or "Matern5_2"
- ZEROMEAN- logical, whether the emulator mean is zero or not. A zero mean corresponds to ZEROMEAN=TRUE and q=0.
- SLEEP- additional argument, set to a zero-vector. This indicates at which iteration to suspend execution of R expressions. This is useful if you are running the sampler for a large (>1e4) number of iterations on a machine which is prone to over-heating.
- filename- additional argument, set to NULL. If not NULL, the output will be saved at every iteration in an RDS file using the name provided (should include ".rds"). This is not recommended to for routine use as it slows down the execution.

The function output is a list with the following elements

1. Results of the MCMC
2. Tuned variances of $\beta$
3. Tuned variance of $\sigma^2$
4. Tuned variances of $\theta$
5. Tuned variance of $\tau$
6. Description

**Figure 1: prior and posterior density plots**

```
#Re-name the object for ease of working
FULL <- FULLsqM5Ntest

#Alternatively, upload the file below
#FULL <- readRDS("FULLsqM5N.rds")
#ADAPTIVE <- 1e5 #The "FULLsqM5N.rds" has 1e5 adaptive/burn-in iterations.

#Remove the burn-in
FULL[[1]] <- FULL[[1]][-(1:ADAPTIVE),]

#Define a vector of prior means
MEANS=c(rep(0,6),11.11111,rep(5,5),0.5)

par(mfrow=c(3,5),mar=c(2,2,2,0.5))
for(i in c(1:6)){
  plot(density(FULL[[1]][,i]),type="l",bty="n",lwd=2,main="")
  abline(v=mean(FULL[[1]][,i]),lwd=2,lty=2)
  abline(v=MEANS[i],lwd=2,lty=2,col=2)
  title(main=bquote(beta[.(i-1)]),line=1,font.lab=2)
  if(i!=1){lines(seq(-20,30,by=0.01),dnorm(seq(-20,30,by=0.01),0,4),col=2,lwd=2)}else{
    lines(seq(-30,30,by=0.01),dnorm(seq(-30,30,by=0.01),0,10),col=2,lwd=2)}
}
i=7;plot(density(FULL[[1]][,i]),type="l",bty="n",lwd=2,main="")
alpha <- 10;beta <-100
abline(v=mean(FULL[[1]][,i]),lwd=2,lty=2);abline(v=MEANS[i],lwd=2,lty=2,col=2)
title(main=expression(sigma^2),line=1,font.lab=2)
lines(seq(0,40,by=0.01), dinvgamma(seq(0,40,by=0.01),alpha,beta),col=2,lwd=2)
for(i in c(8:12)){
  YLIM=max(density(FULL[[1]][,i])$y,dexp(seq(-5,30,by=0.01),0.2))
  plot(density(FULL[[1]][,i]),type="l",bty="n",lwd=2,main="",ylim=c(0,YLIM))
  abline(v=mean(FULL[[1]][,i]),lwd=2,lty=2)
  abline(v=MEANS[i],lwd=2,lty=2,col=2)
  title(main=bquote(theta[.(i-7)]),line=1,font.lab=2)
  lines(seq(-5,30,by=0.01),dexp(seq(-5,30,by=0.01),0.2),col=2,lwd=2)
}
i=13;plot(density(FULL[[1]][,i]),type="l",bty="n",lwd=2,main="")
alpha <- 3;beta <- 1
abline(v=mean(FULL[[1]][,i]),lwd=2,lty=2);abline(v=MEANS[i],lwd=2,lty=2,col=2)
title(main=expression(tau),line=1,font.lab=2)
lines(seq(0,40,by=0.01),dinvgamma(seq(0,40,by=0.01),alpha,beta),col=2,lwd=2)
empty(1,1)
legend("topright",lty=c(1,1,2,2),lwd=2,col=c(1,2,1,2),y.intersp=0.8,
       legend=c("Post dens","Prior dens","Post mean","Prior mean"),bty="n")
```

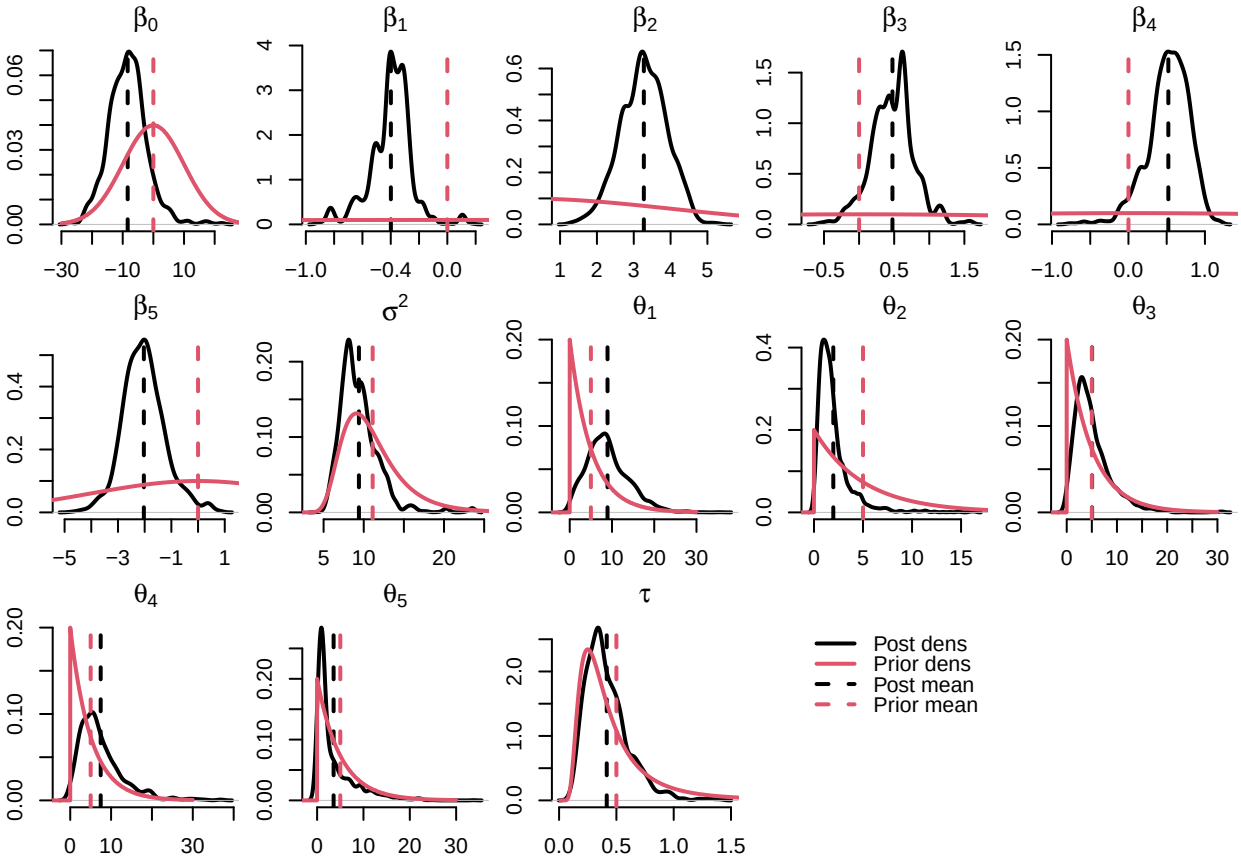
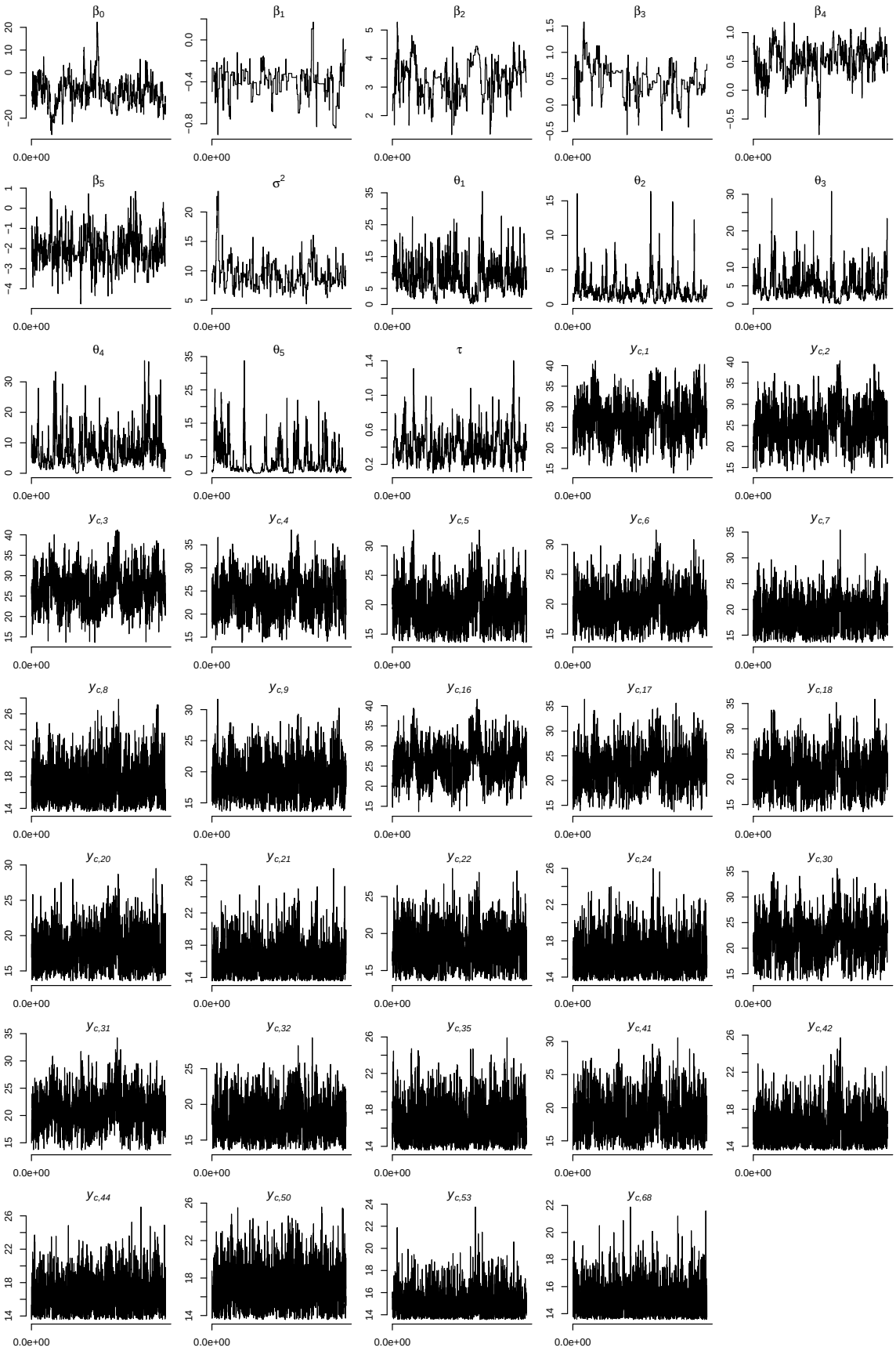



**Figure S4: posterior trace plots**

```
par(mfrow=c(8,5),mar=c(2,2,2,1))
for(i in c(1:6)){
  plot(FULL[[1]][,i],type="l",bty="n",main="",xaxt="n")#,ylim=YLIM
  title(main=bquote(beta[.(i-1)]),line=0.5,font.lab=2)
  axis(1,at=seq(0,2e5,by=5e4),
  labels=formatC(seq(0,2e5,by=5e4), format = "e", digits = 1))
}
i=7;plot(FULL[[1]][,i],type="l",bty="n",main="",xaxt="n")#,ylim=YLIM
title(main=expression(sigma^2),line=0.5,font.lab=2)
axis(1,at=seq(0,2e5,by=5e4),
labels=formatC(seq(0,2e5,by=5e4), format = "e", digits = 1))
for(i in c(8:12)){
  plot(FULL[[1]][,i],type="l",bty="n",main="",xaxt="n")#,ylim=YLIM
  title(main=bquote(theta[.(i-7)]),line=0.5,font.lab=2)
  axis(1,at=seq(0,2e5,by=5e4),
  labels=formatC(seq(0,2e5,by=5e4), format = "e", digits = 1))
}
i=13;plot(FULL[[1]][,i],type="l",bty="n",main="",xaxt="n")#,ylim=YLIM
title(main=expression(tau),line=0.5,font.lab=2)
axis(1,at=seq(0,2e5,by=5e4),
labels=formatC(seq(0,2e5,by=5e4), format = "e", digits = 1))
for(i in seq(14,39)){
```

```
  plot(FULL[[1]][,i],type="l",bty="n",main="",xaxt="n")
  title(main=bquote(italic(y[.(paste(c("c,",sub("Cen","",
  names(FULL[[1]])[i])),sep="",collapse=""))])),font=2,
  line=0.5,font.lab=4)
  axis(1,at=seq(0,2e5,by=5e4),
  labels=formatC(seq(0,2e5,by=5e4), format = "e", digits = 1))
}
```


β0
β1
β2
β3
β4
β5
σ2
θ1
θ2
θ3
θ4
θ5
τ
yc,1
yc,2
yc,3
yc,4
yc,5
yc,6
yc,7
yc,8
yc,9
yc,16
yc,17
yc,18
yc,20
yc,21
yc,22
yc,24
yc,30
yc,31
yc,32
yc,35
yc,41
yc,42
yc,44
yc,50
yc,53
yc,68
0.0e+00