Supplement of Nat. Hazards Earth Syst. Sci., 22, 2611–2635, 2022
https://doi.org/10.5194/nhess-22-2611-2022-supplement

Natural Hazards and Earth System Sciences
Open Access

Supplement of

Introducing SlideforMAP: a probabilistic finite slope approach for modelling shallow-landslide probability in forested situations

Feiko Bernard van Zadelhoff et al.

Correspondence to: Feiko Bernard van Zadelhoff (feiko.vanzadelhoff@bfh.ch)

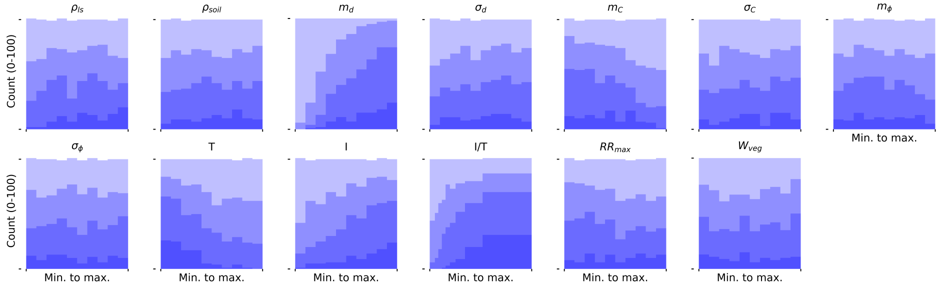

Figure S 1. Histograms of different subsamples of the LHS parameter sets for the Eriz study area. The shading (from light to dark) corresponds to subsamples retaining only the $x\%$ best parameter sets in terms of AUC; the shown fractions are: 1, 0.7, 0.4, 0.1.

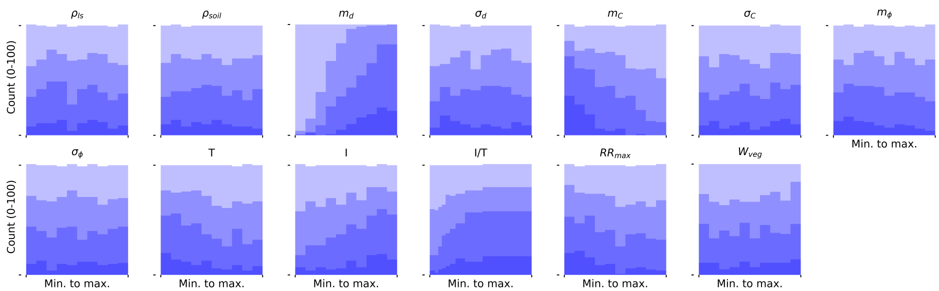

Figure S 2. Histograms of different subsamples of the LHS parameter sets for the Eriz study area. The shading (from light to dark) corresponds to subsamples retaining only the $x\%$ best parameter sets in terms of Unstable ratio; the shown fractions are: 1, 0.7, 0.4, 0.1.

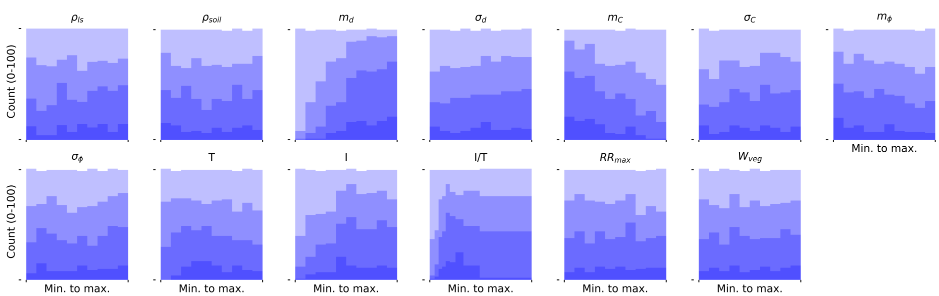

Figure S 3. Histograms of different subsamples of the LHS parameter sets for the StA study area. The shading (from light to dark) corresponds to subsamples retaining only the $x\%$ best parameter sets in terms of AUC; the shown fractions are: 1, 0.7, 0.4, 0.1.

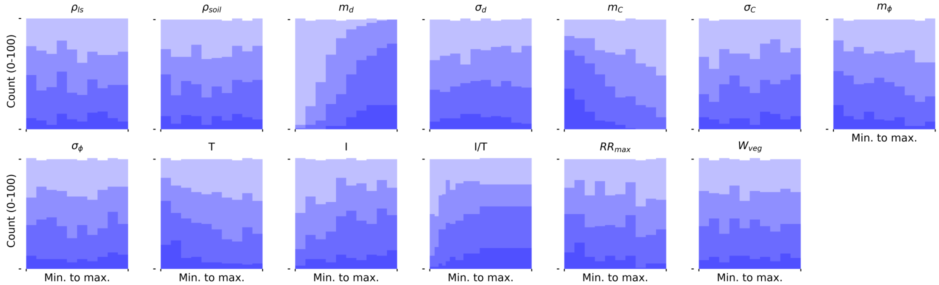

Figure S 4. Histograms of different subsamples of the LHS parameter sets for the StA study area. The shading (from light to dark) corresponds to subsamples retaining only the $x\%$ best parameter sets in terms of Unstable ratio; the shown fractions are: 1, 0.7, 0.4, 0.1.

R-code constructing the precipitation intensity boundaries in the sensitivity analysis:

```
## Computation of the DDF curves ##
## Input intensities manually read from https://hydromaps.ch ->
Wasser in der Atmosphaere ->
B04 Extreme Punktniederschlaege
## method according to: https://hydrologischeratlas.ch/downloads
/01/content/Text_Tafel24.de.pdf

#StAntonien rainfall intensity
#i_1h_100J<-42      #mm/h, 1 hour time period intensity, 100 year return period
#i_24h_100J<-115/24 #mm/h, 24 hour time period, 100 year return period
#i_1h_2p33J<- 21    #mm/h
#i_24h_2p33J<-57/24  #mm/h

#Bern_mittelland (Trub and Eriz study area) rainfall intensity
i_1h_100J<-47      #mm/h
i_24h_100J<-115/24 #mm/h
i_1h_2p33J<- 21    #mm/h
i_24h_2p33J<-62/24  #mm/h

T<-100 # Return period
d<- 1 # Duration
a<-0.315*log(i_24h_100J*24/i_1h_100J) # parameter 1
b<-0.315*log(i_24h_2p33J*24/i_1h_2p33J) # parameter 1

y<- -log(-log(1-1/T))
i_d_T_1<- i_1h_2p33J*(d^b) + 0.248*(i_1h_100J*(d^a) - i_1h_2p33J*(d^b))*(y-0.577)
print(i_d_T_1)
```

Python-code make and validating a log-normal distribution from a mean and a standard deviation:

```
import numpy as np
import scipy.special as sps
import matplotlib.pyplot as plt
from sklearn.metrics import mean_squared_error
from scipy.stats import norm
import pandas as pd

######### NORMAL TO LOGNORMAL DISTRIBUTION #########
LSFile = ’LS_data.csv’
LSinv_base = pd.read_csv(LSFile)
soil_depth_obs = LSinv_base[’depth’].dropna().tolist() # get the list of the
```

```
#https://blogs.sas.com/content/iml/2014/06/04/simulate-lognormal-data-with-specified-mean-and-variance.
    html
m = np.mean(soil_depth_obs) # mean
v = np.std(soil_depth_obs)**2 # variance
phi = np.sqrt(v + m**2)
mu = np.log(m**2/phi) # lognormal par 1
sigma = np.sqrt(np.log(phi**2/m**2)) # lognormal par 2
n = 1000000

normalVals = np.random.normal(m,np.sqrt(v),n)                           # draw soil depths from the
    distribution
LognormalVals = np.exp(np.random.normal(mu,sigma,n))            # draw soil depths from the distribution

plt.hist([normalVals, LognormalVals],bins = 500, histtype= 'step')
plt.xlabel('soil_depth')
plt.ylabel('n')

print(len(normalVals))
print(len(LognormalVals))
print(np.mean(normalVals))
print(np.mean(LognormalVals))
print(np.min(normalVals))
print(np.min(LognormalVals))
print(np.max(normalVals))
print(np.max(LognormalVals))
print(np.std(normalVals))
print(np.std(LognormalVals))

######## SOIL DEPTH OUR DISTRIBUTIONS VS. DATA #############
x = np.linspace(-1, 4, 5000)
pdf_log = (np.exp(-(np.log(x) - mu)**2 / (2 * sigma**2)) / (x * sigma * np.sqrt(2 * np.pi)))

points = plt.hist(soil_depth_obs, bins=int((max(soil_depth_obs)-min(soil_depth_obs))/0.2), density =
    True, label = 'normalized data (n = 607)')
plt.plot(x, norm.pdf(x, m, np.sqrt(v)), label = 'fit normal distribution')
plt.plot(x, pdf_log, linewidth=2, color='r', label = 'fit lognormal distribution')
plt.axis('tight')
plt.xlabel('soil thickness (m)')
plt.ylabel('probability density')
plt.legend()
plt.savefig(r'C:\Users\vof1\Documents\Papers\paper_1\reviews\review_2\response\normallognormal.png',
    bbox_inches='tight', dpi=200)
plt.show()

######## COMPUTE MEAN SQUARE ROOT ERRORS #############
LS_x = np.linspace(0.1, 2.9, 15)
LS_lognormals = np.nan_to_num((np.exp(-(np.log(LS_x) - mu)**2 / (2 * sigma**2)) / (LS_x * sigma * np.
    sqrt(2 * np.pi))))
LS_normals = norm.pdf(LS_x, m, np.sqrt(v))

# Mean Squared Error
MSE_normal = np.square(np.subtract(points[0],LS_normals)).mean()
MSE_lognormal = np.square(np.subtract(points[0],LS_lognormals)).mean()

print('The normal distribution MSE is ', np.round(MSE_normal,4))
print('The lognormal distribution MSE is ', np.round(MSE_lognormal,4))
```

R-code computing and plotting RRmax, Lateral root reinforcement and Basal root reinforcement:

```
#Data from Gehring et al. 2019. (https://doi.org/10.1038/s41598-019-45073-7) and applied SlideforMap
```

```
Distance <- seq(0.1,15,0.1)    # Range of horizontal distance [m]
depth<- seq(0,2,0.02)          # Range of soil depth [m]
DBH <-0.3                      # Diameter at Breast height of a tree [m]

# Horizontal root density Gamma function parameters
alpha_1 <- 0.862
beta_1 <- 3.225
c <- 25068.54

# Vertical root density Gamma function parameters
alpha_2 <- 1.284
beta_2 <- 3.688

# Check the maximum root reinforcement with alpha_1 and beta_1
root_max <- ifelse(Distance <18.5, ((c*DBH)*dgamma(Distance/(DBH*18.5), alpha_1, beta_1)),0)  #[N/m],
plot(Distance,root_max/1000, type = "b", pch = 16, col = '#FF8C00', main = 'RRmax,␣DBH␣=␣0.3␣m.', xlab = 'Distance␣from␣stem␣[m]', ylab = "Root␣reinforcement␣[kN/m]")
grid()

# lateral root reinforcement and basal root reinforcement with alpha_2 and beta_2
k <- 1 # [m^(-1)]
root_max_fix <- 1 # [kN/m]
degamma <- dgamma(depth,alpha_2,beta_2)
RRlat <- root_max_fix*pgamma(depth,alpha_2,beta_2) # [kN/m]
RRbas <- k * dgamma(depth,alpha_2,beta_2)*root_max_fix  # [kN/m^2]

plot(depth,degamma, type = "b", pch = 16, col = '#838B83', xlab = 'Soil␣thickness␣[m]', ylab = "Root␣reinforcement")
lines(depth, RRlat, type = "b", pch = 16, col = '#FF8C00')
lines(depth, RRbas, type = "b", pch = 16, col = '#1E90FF')
grid()
legend("topright",legend = c("Lateral␣[kN/m]", "Basal␣[kN/m2]"), col = c("#FF8C00", "#1E90FF"),pch = c(16, 16))

# Plot for an actual Random landslide
Surface_area = 50 # [m^2]
length_width_ratio <- 2 # [-]
ls_width = sqrt((Surface_area*4)/(pi*length_width_ratio))       # [m]
ls_length = ls_width*length_width_ratio  # [m]
Circumference = pi*(3*((ls_length/2)+(ls_width/2))-sqrt((3*(ls_length/2)+(ls_width/2))*((ls_length/2)+(3*(ls_width/2))))) # [m]

RRlat_l <- root_max_fix * pgamma(depth,alpha_2,beta_2) * Circumference * 0.5  # kN
RRbas_l <- root_max_fix * dgamma(depth,alpha_2,beta_2) * Surface_area         # kN

plot(depth,RRbas_l, type = "b", pch = 16, col = '#1E90FF', main = 'Random␣Landslide,␣area␣=␣50␣m2', xlab = 'Soil␣thickness␣[m]', ylab = "Root␣Force␣[kN]")
lines(depth, RRlat_l, type = "b", pch = 16, col = '#FF8C00')
grid()
legend("topright",legend = c("Lateral", "Basal"), col = c("#FF8C00", "#1E90FF"),pch = c(16, 16))
```