```r
getwd()
soil = read.csv("soilresptemp.txt", sep = " ")

Rsoil = soil$Rsoil
Tsoil = soil$Tsoil

Rsoil_rmv = Rsoil[Rsoil > 0]
Tsoil_rmv = Tsoil[Rsoil > 0]


loglik = function(theta, x, y){
  gamma = theta[1:2]
  beta = theta[3]
  n = length(y)
  eta = x %*% gamma
  mu = exp(eta)
  alpha = mu * beta
  return(sum(sapply(1:n, function(i) dgamma(y[i], shape = alpha[i], rate = beta,  log = TRUE))))
}

loglik_neg = function(theta, x, y){
  return(-loglik(theta, x, y))
}

grad = function(theta, x, y){
  gamma = theta[1:2]
  beta = theta[3]
  eta = x %*% gamma
  mu = exp(eta)
  alpha = mu * beta

  l_alpha = log(beta) - digamma(alpha) + log(y)
  l_beta = sum(mu - y)
  l_gamma = beta * t(x) %*% (l_alpha * mu)

  return(c(l_gamma, l_beta))
}


getest2 = function(x, y, theta_ini, ..., eps = 10^(-5)){
  gamma_ini = theta_ini[1:2]
  beta_ini = theta_ini[3]
  iter = 0
  theta = c(gamma_ini, beta_ini)

  while(TRUE){
    gamma = theta[1:2]
    beta = theta[3]
    iter = iter + 1
    eta = x %*% gamma
    mu = exp(eta)
    alpha = mu * beta
    l_alpha = log(beta) - digamma(alpha) + log(y)
    l_beta = sum(mu - y)
    l_gamma = beta * t(x) %*% (l_alpha * mu)

    l_theta = c(l_gamma, l_beta)

    l_alpha2 = - trigamma(alpha)
    l_beta2 = -sum(alpha^2 / beta)


    w11 = l_alpha2 * mu^2 * beta^2 + l_alpha * mu * beta

    l_gamma2 = t(x) %*% diag(c(w11)) %*% x
    l_gammabeta = t(x) %*% mu

    l_theta2 = rbind(cbind(l_gamma2, l_gammabeta), c(l_gammabeta, l_beta2))

    #print(iter)
    #print(l_theta)
    #print(l_theta2)

    theta_2 = c(gamma, beta) - solve(l_theta2, l_theta)

    logliktmp = loglik(theta, x, y)
    logliktmp2 = loglik(theta_2, x, y)

    if(any(is.na(theta_2))){
      stop("Fatal error:: NA's generated")
    }

    if(norm(theta - theta_2, "2") < eps & norm(logliktmp - logliktmp2, "2") < eps){
      theta = theta_2
      break
    }
    else{
      theta =  theta_2
    }
  }
  gamma = theta_2[1:2]
  beta = theta_2[3]
  logliktmp = loglik(theta, x, y)

  return(list(gamma = gamma, beta = beta, iteration = iter, gradient = l_theta,
              loglik = logliktmp, info = -l_theta2))
}

#prob 1
x = cbind(1, Tsoil_rmv)
y = Rsoil_rmv

model_ini = glm(Rsoil_rmv ~ Tsoil_rmv, family = Gamma(link = "log"))
summary(model_ini)
```

```r
plot(residuals(model_ini, type = "deviance") ~ model_ini$fitted.values,
     xlab = "Fitted Value", ylab = "Deviance Residual")
abline(h = 0)

smodel_ini <- summary(model_ini)
alpha_tmp <- 1 / smodel_ini$dispersion
beta_tmp <- alpha_tmp / model_ini$fitted.values
#
# r = sapply(1:length(y), function(k) pgamma(y[k], shape = alpha_tmp, rate = beta_tmp[k]))
# plot(r ~ mu)
# sum(((y - mu) / sqrt(mu^2 * smodel_ini$dispersion))^2)
y_random = sapply(1:length(y), function(k) rgamma(1, shape = alpha_tmp, rate = beta_tmp[k]))
plot(y_random ~ x[,2])
plot(y ~ x[,2])


#prob 2, 3
gamma_ini = model_ini$coefficients
beta_ini = 1 / mean((y - model_ini$fitted.values)^2 / model_ini$fitted.values)
theta_ini = c(gamma_ini, beta_ini)

res = optim(par = theta_ini, fn = loglik_neg, x = x, y = y)
res$par
loglik(res$par, x, y)

#optim(par = theta_ini, fn = loglik_neg, gr = grad, x = x, y = y, method = "L-BFGS-B",
#      upper = c(Inf, Inf, Inf), lower = c(-Inf, -Inf, 10^(-20)))

#optim(par = theta_ini, fn = loglik_neg, x = x, y = y, method = "L-BFGS-B",
#       upper = c(Inf, Inf, Inf), lower = c(-Inf, -Inf, 10^(-20)))

res2 = getest2(x, y, res$par, eps = 10^(-6))
loglik(c(res2$gamma, res2$beta), x, y)

loglik(res$par, x, y) < loglik(c(res2$gamma, res2$beta), x, y)
gamma = res2$gamma
beta = res2$beta

mu = exp(x %*% gamma)

alpha = mu * beta
#r = sapply(1:length(y), function(k) pgamma(y[k], shape = alpha[k], rate = beta))
#plot(r ~ mu)
plot((y - mu) / sqrt(mu / beta) ~ mu)

y_random = sapply(1:length(y), function(k) rgamma(1, shape = alpha[k], rate = beta))
plot(y_random ~ x[,2], xlab = "x", ylab = "y", main = "simulated realizations")
plot(y ~ x[,2], xlab = "x", ylab = "y", main = "observations")

# Prob 4 and 5
x = cbind(1, Tsoil)
y = Rsoil
```

```
#install.packages("extRemes")
library(extRemes)
loglik_evd1 = function(theta, x, y, ..., phi = 1){
  beta = theta[1:2]
  sigma = theta[3]
  n = length(y)
  eta = x %*% beta
  xi = exp(eta)
  theta = sigma * xi ^ phi
  return(sum(sapply(1:n, function(i) devd(y[i], loc = xi[i], scale = theta[i], log = TRUE))))
}

loglik_evd_neg1 = function(theta, x, y, ..., phi = 1){
  return(-loglik_evd1(theta, x, y, phi = phi))
}

beta_ini = gamma_ini
sigma_ini = sqrt(6 / beta) / pi

theta_ini = c(beta_ini, sigma_ini)
loglik_evd1(theta_ini, x, y, phi = 1)

res = optim(par = theta_ini, fn = loglik_evd_neg1, x = x, y = y)
res$par
beta = res$par[1:2]
sigma = res$par[3]
loglik_evd1(res$par, x, y)

#res = optim(par = c(gamma, beta), fn = loglik_evd_neg1, x = x, y = y)
#res$par
#loglik_evd1(res$par, x, y)

eta = x %*% beta
xi = exp(eta)
phi = 1
theta = sigma * xi ^ phi
plot((y - xi) / sqrt(pi^2 * theta^2 / 6) ~ xi)

y_random = sapply(1:length(y), function(k) revd(1, loc = xi[k], scale = theta[k]))
plot(y_random ~ x[,2], xlab = "x", ylab = "y", main = "simulated realizations")
plot(y ~ x[,2], xlab = "x", ylab = "y", main = "observations")

# Prob 6
loglik_evd2 = function(theta, x, y){
  beta = theta[1:2]
  sigma = theta[3]
  phi = theta[4]
  n = length(y)
  eta = x %*% beta
  xi = exp(eta)
  theta = sigma * xi ^ phi
  return(sum(sapply(1:n, function(i) devd(y[i], loc = xi[i], scale = theta[i], log = TRUE))))
}
```

```r
loglik_evd_neg2 = function(theta, x, y){
  return(-loglik_evd2(theta, x, y))
}

beta_ini = beta
sigma_ini = sigma
phi_ini = 1

theta_ini = c(beta_ini, sigma_ini, phi_ini)

res = optim(par = theta_ini, fn = loglik_evd_neg2, x = x, y = y)
res$par
beta = res$par[1:2]
sigma = res$par[3]
phi = res$par[4]
loglik_evd2(res$par, x, y)

phi_vec = seq(0.2, 3, by = 0.2)
lik_vec = sapply(phi_vec, function(k) optim(par = theta_ini, fn = loglik_evd_neg1,
                                            x = x, y = y, phi = k)$value)

plot(phi_vec, lik_vec, type = "l", xlab = "phi", ylab = "negative log-likelihood")

eta = x %*% beta
xi = exp(eta)
theta = sigma * xi ^ phi
plot((y - xi) / sqrt(pi^2 * theta^2 / 6) ~ xi)
sum(((y - xi) / sqrt(pi^2 * theta^2 / 6))^2)
plot((y - xi) / theta  ~ xi)
r = sapply(1:length(y), function(k) pevd(y[k], loc = xi[k], scale = theta[k]))
plot(r ~ xi)

# Prob 7
y_random = sapply(1:length(y), function(k) revd(1, loc = xi[k], scale = theta[k]))
plot(y_random ~ x[,2], xlab = "x", ylab = "y", main = "simulated realizations")
plot(y ~ x[,2], xlab = "x", ylab = "y", main = "observations")
```