Let's assume that we have a series system with $m = 3$ components, each of which has an exponentially distributed time-to-failure,

$$T_{ij} \sim \text{exponential}(\lambda_j)$$

for $j = 1, \ldots, m$ and $i = 1, \ldots, n$ where $\boldsymbol{\theta} = (\lambda_1, \lambda_2, \lambda_3)$ is unknown.

We replicate the masked data in Table 2 from the Guo paper:

```
md <- read.csv("./data.csv")
md$X1 <- ifelse(md$X1 == 0,F,T)
md$X2 <- ifelse(md$X2 == 0,F,T)
md$X3 <- ifelse(md$X3 == 0,F,T)
md
```

```
##    System.ID Failure.Time    X1    X2    X3
## 1          1           21 FALSE  TRUE FALSE
## 2          2           38  TRUE  TRUE FALSE
## 3          3           54 FALSE FALSE  TRUE
## 4          4           66 FALSE FALSE  TRUE
## 5          5           76  TRUE  TRUE FALSE
## 6          6           78 FALSE  TRUE  TRUE
## 7          7          123 FALSE FALSE  TRUE
## 8          8          130  TRUE FALSE  TRUE
## 9          9          152  TRUE  TRUE  TRUE
## 10        10          159  TRUE FALSE FALSE
## 11        11          199 FALSE FALSE  TRUE
## 12        12          201  TRUE FALSE FALSE
## 13        13          204  TRUE FALSE FALSE
## 14        14          215 FALSE  TRUE  TRUE
## 15        15          218  TRUE  TRUE FALSE
## 16        16          281  TRUE FALSE FALSE
## 17        17          295 FALSE  TRUE FALSE
## 18        18          310 FALSE FALSE  TRUE
## 19        19          338 FALSE FALSE  TRUE
## 20        20          341 FALSE  TRUE FALSE
## 21        21          354  TRUE FALSE FALSE
## 22        22          358 FALSE  TRUE FALSE
## 23        23          431  TRUE  TRUE  TRUE
## 24        24          457 FALSE FALSE  TRUE
## 25        25          545  TRUE  TRUE  TRUE
## 26        26          569 FALSE  TRUE FALSE
## 27        27          677 FALSE FALSE  TRUE
## 28        28          818 FALSE  TRUE FALSE
## 29        29          946 FALSE  TRUE  TRUE
## 30        30         1486  TRUE FALSE FALSE
```

We are using the likelihood function given by

$$\begin{aligned}
L(\lambda_1, \lambda_2, \lambda_3) &= \prod_{i=1}^{n} \sum_{j \in C_i} f_j(t_i; \theta_j) \prod_{\substack{p=1 \\ p \neq j}} R_p(t_i; \theta_p) \\
&= \prod_{i=1}^{n} \left[\left\{ \prod_{j=1}^{m} R_j(t_i; \theta_j) \right\} \left\{ \sum_{k \in C_i} h_k(t_i; \theta_k) \right\} \right]
\end{aligned}$$

and the log-likelihood function given by

$$l(\lambda_1, \lambda_2, \lambda_3) = \sum_{i=1}^{n} \sum_{j=1}^{m} \log R_j(t_i; \theta_j) + \sum_{i=1}^{n} \log \left\{ \sum_{k \in C_i} h_k(t_i; \theta_k) \right\}.$$

The component times-to-failure are exponentially distributed, and thus the hazard and survival functions are respectively given by $h_j(t; \lambda_j) = \lambda_j$ and $R_j(t; \lambda_j) = \exp(-\lambda_j t)$. Making this substitution into the likelihood function obtains the result

$$L(\lambda_1, \lambda_2, \lambda_3) = \prod_{i=1}^{n} (\sum_{j \in C_i} \lambda_j) e^{-\left(\sum_{j=1}^{m} \lambda_j\right) t_i}. \tag{1}$$

and into the log-likelihood function obtains the result

$$\ell(\lambda_1, \lambda_2, \lambda_3) = \sum_{i=1}^{n} \log\big(\sum_{j \in C_i} \lambda_j\big) - \big(\sum_{j=1}^{m} \lambda_j\big)\big(\sum_{i=1}^{n} t_i\big). \tag{2}$$

We model the log-likelihood function in R with:

```r
# generator for log-likelihood function
loglike <- function(ttf,cand)
{
    n <- length(ttf)
    function(theta)
    {
        res <- 0
        for (i in 1:n)
        {
            #cat("i=",i,"ttf=",ttf[i],"C=",(1:m)[cand[i,]],"sum=",sum(theta[cand[i,]]),"\n")
            res <- res + log(sum(theta[cand[i,]]))
        }
        return(res - sum(theta) * sum(ttf))
    }
}

mle.grad <- function(l,theta0,eps=1e-3,r=.5)
{
    repeat
    {
        # we use backtracking for an approximate line search
        alpha <- 1
        theta1 <- NULL
        g <- numDeriv::grad(l,theta0)
        repeat {
            theta1 <- theta0 + alpha * g
            if (all(theta1 > 0) && l(theta1) > l(theta0))
                break
            alpha <- r*alpha
        }

        # infinity norm
        if (abs(sum(theta1-theta0)) < eps)
            return(theta1)
        theta0 <- theta1
    }
}
```

So, let's set up the log-likelihood function:

```
ttf <- md$Failure.Time
cand <- as.matrix(md[3:5])
l <- loglike(ttf,cand)
```

Now, we try to solve the MLE using gradient ascent to solve for the zeros of the gradient of the log-likelihood function. We repeat the method many times, using random initial points, and choose the best one found:

```
trials <- 1
theta.hat <- NULL
l.theta.hat <- -Inf
repeat
{
  theta0 <- runif(3,1.5,6.5)
  theta.b <- NULL
  tryCatch(
    {
      theta.b <- mle.grad(l,theta0,1e-6)
      l.theta.b <- l(theta.b)
      if (l.theta.b > l.theta.hat)
      {
        l.theta.hat <- l.theta.b
        theta.hat <- theta.b
      }
      trials <- trials + 1L
    },
    warning = function(w) {},
    error = function(e) {})

  if (trials == 5000L)
    break
}

print(theta.hat)
## [1] 0.0008579827 0.0009880395 0.0011126141
```

So, we see that $\hat{\theta} = (0.0008579827, 0.0009880395, 0.0011126141)$.