Journal of the Chinese
Statistical Association
Vol. 57, (2019) 1–42

The COM-Poisson Cure Rate Model for Survival Data-Computational Aspects

Zhisheng He[1] **and Takeshi Emura**[2,†]

[1]Institute of Materia Medica, Zhejiang Academy of Medical Sciences, China
[2]Graduate Institute of Statistics, National Central University

ABSTRACT

The Conway-Maxwell-Poisson (COM-Poisson) distribution is useful to account for a cure proportion in survival data. With this model, two computational approaches for calculating maximum likelihood estimates have been developed in the literature: one based on the method in the gamlss R package that employs the first-order derivatives of the log-likelihood, and the other based on the EM algorithm that employs the complete-data likelihood. In this paper, we propose a robust version of the Newton-Raphson (NR) algorithm, where the robustness is introduced by random perturbations to the initial values and by log-transformations to positive parameters. We provide the expressions of the derivatives of the log-likelihood under the Bernoulli cure model and computer codes for implementation. Since the NR algorithm employs the first- and second-derivatives of the log-likelihood, it converges more quickly than the method of the gamlss R package. We also review the EM algorithms and compare the computational performance between the NR and EM algorithms via simulations. We also include a novel data to be fitted to the COM-Poisson cure model, and discuss the consequence of performing the two algorithms.

Key words and phrases: EM algorithm, Generalized gamma distribution, Newton-Raphson algorithm, Survival analysis, Weibull distribution.

JEL classification: C13, C46.

[†]Corresponding to: Takeshi Emura
E-mail: takeshiemura@gmail.com

1. Introduction

Cure rate models are models for analyzing survival data containing a cured proportion. For instance, during the treatment of a disease, a patient can be cured, which means he/she shows no recurrence of the disease. In such a case, researchers are interested in how covariates (e.g., treatment type and gender) influence the occurrence of cure.

Patients who do not experience recurrence of a disease are called *long-term survivors* (Maller and Zhou 1996). Patients who experience the recurrence of a disease are called *susceptibles*. Standard statistical models for survival analysis assume that all the patients are susceptibles after a long follow-up, which mean that they eventually experience a failure event. This assumption is often not correct.

In the theory of competing risks, event time of a subject is determined by several different causes of the event (Cox and Oakes 1984). This implies that the observed event time of a subject is determined by the first-occurring cause among several different causes. In this respect, Rodrigues et al. (2009) adopted a Conway-Maxwell Poisson (COM-Poisson) distribution to describe the number of causes, where the occurrence of cure is accounted by the zero value of the distribution. Note that the Bernoulli distribution is a special case of the COM-Poisson distribution, representing the binary (cure vs. non-cure) setting.

Lifetime models under the COM-Poisson cure model require some parametric models for failure times for all the causes. Rodrigues et al. (2009) and Balakrishnan and Pal (2016) suggested the Weibull distribution. Balakrishnan and Pal (2015) suggested the generalized gamma (G-gamma) distribution that includes the Weibull, lognormal, and gamma distributions as special cases.

Different computational algorithms exist for finding the maximum likelihood estimator (MLE) under the COM-Poisson cure model. Rodrigues et al. (2009) and de Castro et al. (2010) proposed an iteration algorithm based on the *gamlss* R package, to maximize the log-likelihood by utilizing its first-order derivatives. However, such an approach typically needs a larger number of iterations than approaches that utilize both the first- and second-order derivatives. Balakrishnan and Pal (2015, 2016) developed

the EM algorithm that utilizes the first- and second-order derivatives. However, EM algorithms are complicated algorithms that iterate between E-step and M-step, where M-step has "inside" iterations.

In general, the convergence speed (say, the number of iterations) of the EM algorithm is slower than the Newton-Raphson (NR) algorithm. On the other hand, the EM algorithm is less sensitive to the initial values than the NR algorithm. Such comparative studies between the EM and NR algorithms can be found in a variety of contexts (e.g., MacDonald 2014; Meng 2014; Emura and Shiu 2016). However, the study of this type has not been conducted under the COM-Poisson cure lifetime models since the NR algorithm has not been considered. The reason might be that the NR algorithm is sensitive to the initial values.

In this paper, we develop a robust version of the NR algorithm and compare the two computational algorithms (NR and EM) under the COM-Poisson cure model. We derive the first- and second-order derivatives of the log-likelihood for the NR algorithm under the Bernoulli cure model, a special case of the COM-Poisson cure model. We compare the convergence properties between the NR and EM algorithms via simulations. We also compare the two algorithms through the analysis of patients with uveal cancer that are new data in the context of cure models. We provide the R codes to implement the proposed NR algorithm and conduct simulations.

The paper is organized as follows. Section 2 reviews the COM-Poisson cure model. Section 3 introduces the likelihood, MLE, and EM algorithm. Section 4 develops the NR algorithm. Section 5 conducts simulations and Section 6 analyzes real data. Section 7 concludes. Appendices include the detailed mathematical expressions and R codes.

2. COM-Poisson cure model for survival data

This section reviews the cure model for survival data based on the COM-Poisson distribution.

2.1 The Conway-Maxwell-Poisson (COM-Poisson) distribution

The Conway-Maxwell-Poisson (COM-Poisson) distribution is a family of discrete distributions introduced by Conway and Maxwell (1962). Let M be the number of failure causes following the COM-Poisson distribution

$$P(M = m; \eta, \phi) = \frac{1}{Z(\eta, \phi)} \frac{\eta^m}{(m!)^\phi}, \ m = 0, 1, 2, \ldots; \ \eta > 0, \ \phi \geq 0,$$

where

$$Z(\eta, \phi) = \sum_{j=0}^{\infty} \frac{\eta^j}{(j!)^\phi}.$$

If the preceding sum is infinite, the distribution is undefined. For instance, when $\phi = 0$, the infinite sum is finite only when $0 < \eta < 1$. When $\phi = 1$, the Poisson distribution is defined for any $\eta > 0$. By setting a proper parameter space for $\eta > 0$ and $\phi \geq 0$, the COM-Poisson distribution yields the three discrete distributions, namely Bernoulli, Poisson, and Geometric distributions (Table 1).

The distributional characteristics (e.g., moments and distribution function) of the COM-Poisson distribution are well-developed (Nadarajah 2009). In addition, applications of the COM-Poisson distribution can be seen in both engineering (Sellers 2012) and biomedical studies.

Table 1: Three different cases of the COM-Poisson distribution.

	Distribution	$Z(\eta, \phi)$	Cured proportion
$\phi = 0$ and $\eta < 1$	Geometric	$\frac{1}{(1-\eta)}$	$p_0 = 1 - \eta$
$\phi = 1$	Poisson	$\exp(\eta)$	$p_0 = \frac{1}{\exp(\eta)}$
$\phi \to \infty$	Bernoulli	$1 + \eta$	$p_0 = \frac{1}{(1+\eta)}$

2.2 Cure proportion

To account for the cured proportion in a population, Rodrigues et al. (2009) assumed that the cure proportion (no failure cause) is

$$p_0 = P(M = 0; \eta, \phi) = \frac{1}{Z(\eta, \phi)}.$$

For instance, the Bernoulli distribution corresponds to

$$\lim_{\phi \to \infty} Z(\eta, \phi) = \lim_{\phi \to \infty} \sum_{j=0}^{\infty} \frac{\eta^j}{(j!)^\phi} = \lim_{\phi \to \infty} \left[\frac{1}{1} + \frac{\eta}{1} + \frac{\eta^2}{(2!)^\phi} + \dots \right] = 1 + \eta.$$

Therefore,

$$p_0 = P(M = 0; \eta, \infty) = \frac{1}{1+\eta}, \quad P(M = 1; \eta, \infty) = \frac{\eta}{1+\eta},$$

This model implies that the population is a mixture of the cured patients and non-cured patients, yielding the traditional cure model, called a mixture cure model (Boag 1949). Hence, the COM-Poisson cure model is a generalization of the traditional cure mdoel.

2.3 Lifetime distributions

Given $M = m$, let $W_j (j = 1, 2, \dots m)$ denote the time-to-event due to the j-th cause of failure. Assume that W_j's are iid, with a common distribution function $F(y) = Pr(W_j \leq y)$ and survival function $S(y) = 1 - F(y)$, and that W_j's and M are independent. The lifetime is defined as the time at which at least one failure causes occur,

$$Y = \min\{W_0, W_1, W_2, \dots, W_M\},$$

where $W_0 = \infty$. This leads to the cure proportion $p_0 = Pr(Y = \infty) = Pr(M = 0)$. Note that, if W_j's follow the exponential distribution, then, the resultant distribution of $Y \mid M > 0$ is called the exponential COM-Poisson distribution (Cordeiro et al. 2012).

Example 1. The Weibull distribution

The Weibull model (Balakrishnan and Pal 2016) is specified as the survival function

$$S(\omega;\gamma) = P(W_j > \omega) = \exp[-(\gamma_2\omega)^{\frac{1}{\gamma_1}}],\ \omega > 0$$

for $\gamma = (\gamma_1, \gamma_2)$, where $\gamma_1 > 0$ is the shape parameter and $\gamma_2 > 0$ is the scale parameter. The probability density function (pdf) is

$$f(\omega;\gamma) = -\frac{d}{d\omega}S(\omega;\gamma) = \frac{1}{\gamma_1\omega}(\gamma_2\omega)^{\frac{1}{\gamma_1}}S(\omega;\gamma),\ \omega > 0.$$

Example 2. The generalized gamma (G-gamma) distribution

The G-gamma model (Balakrishnan and Pal 2015) is specified as the pdf

$$f(\omega;\gamma) = \begin{cases} q\{q^{-2}(\lambda\omega)^{\frac{q}{\sigma}}\}^{q^{-2}}\exp\{-q^{-2}(\lambda\omega)^{\frac{q}{\sigma}}\}/\{\Gamma(q^{-2})\sigma\omega\}, & \text{if } q > 0; \\ (\sqrt{2\pi}\sigma\omega)^{-1}\exp[-\{\log(\lambda\omega)\}^2/(2\sigma^2)], & \text{if } q = 0, \end{cases}$$

for $\gamma = (q, \sigma, \lambda)$, where $q \geq 0$ and $\sigma > 0$ are the shape parameters, and $\lambda > 0$ is the scale parameter.

The case $q = 1$ gives the Weibull model ($\gamma_1 = \sigma$ and $\gamma_2 = \lambda$) with the pdf

$$f(\omega;1,\sigma,\lambda) = \frac{1}{\sigma\omega}(\lambda\omega)^{\frac{1}{\sigma}}\exp\{-(\lambda\omega)^{\frac{1}{\sigma}}\}.$$

The case $q \to 0$ gives the lognormal model with the pdf,

$$f(\omega;0,\sigma,\lambda) = \frac{1}{\sqrt{2\pi}\sigma\omega}\exp\left[-\frac{\{\log(\lambda\omega)\}^2}{2\sigma^2}\right].$$

The case $q = \sigma$ gives the gamma model ($\alpha{=}\sigma^{-2}$ and $\beta = \sigma^2/\lambda$) with the pdf

$$f(\omega;\sigma,\sigma,\lambda) = \frac{1}{\Gamma(\sigma^{-2})}\left(\frac{\lambda}{\sigma^2}\right)^{\sigma^{-2}}(\omega)^{\sigma^{-2}-1}\exp\left(-\frac{\lambda}{\sigma^2}\omega\right).$$

The survival function of the G-gamma distribution is

$$S(\omega;\gamma) = \begin{cases} \Gamma(q^{-2}, q^{-2}(\lambda\omega)^{\frac{q}{\sigma}})/\Gamma(q^{-2}), & \text{if } q > 0; \\ 1 - \Phi\left(\frac{\log(\lambda\omega)}{\sigma}\right), & \text{if } q = 0, \end{cases}$$

where $\Gamma(a,b) = \int_b^\infty x^{a-1}e^{-x}dx$ is the *upper incomplete gamma function* and $\Phi(\cdot)$ is the cumulative distribution function (cdf) of $N(0,1)$.

2.4 Long-term survival function

Rodrigues et al. (2009) showed that the survival function of Y is given by

$$S_p(y) = Pr(Y > y) = \frac{Z(\eta S(y), \phi)}{Z(\eta, \phi)},$$

where $S(y) = Pr(W_j > y)$. Note that $S_p(y)$ is not a proper survival function in the sense of $\lim_{y\to\infty} S_p(y) \neq 0$. For this reason, we call $S_p(y)$ as long-term survival function. Indeed,

$$\lim_{y\to\infty} S_p(y) = \lim_{y\to\infty} \frac{Z(\eta S(y), \phi)}{Z(\eta, \phi)} = \frac{Z(0, \phi)}{Z(\eta, \phi)} = \frac{1}{Z(\eta, \phi)} = p_0 > 0.$$

The last expression is the cured proportion.

The derivative of $S_p(y)$ yields the improper density function of Y,

$$f_p(y) = \frac{1}{Z(\eta, \phi)} \frac{f(y)}{S(y)} \sum_{j=0}^{\infty} \frac{j\{\eta S(y)\}^j}{(j!)^\phi}.$$

One has an alternative expression for $S_p(y)$ by considering the population as a mixture of cured and non-cured populations, namely,

$$\begin{aligned} S_p(y) &= P(M > 0)P(Y > y|M > 0) + P(M = 0)P(Y > y|M = 0) \\ &= (1 - p_0)S_1(y) + p_0, \end{aligned}$$

where $S_1(y) = P(Y > y|M > 0)$ denotes the survival function of the non-cured population. Note that $S_1(y)$is a proper survival function in that $\lim_{y\to\infty} S_1(y) = 0$.

3. Likelihood and MLE

We introduce the likelihood function and define the maximum likelihood estimator (MLE). We also discuss the complete-data likelihood function to be used for the EM algorithm.

3.1 Right-censored data with cure

Let Y_i be the lifetime variable, C_i be the right-censored time, and $T_i = \min\{Y_i, C_i\}$ be the observed lifetime for a subject i. Define the censoring indicator and cure indicator

$$\delta_i = I(Y_i \leq C_i) = \begin{cases} 0 & \text{if the observation is censored,} \\ 1 & \text{if the observation is not censored .} \end{cases} ; \; I_i = \begin{cases} 1 & \text{if not cure,} \\ 0 & \text{if cure.} \end{cases}$$

What we observe are T_i and δ_i. We know I_i=1 when $\delta_i = 1$. However, we are uncertain about I_i when $\delta_i = 0$. We also observe covariates $\mathbf{x}_i' = (1, x_{i1}, \ldots, x_{ik})$, where the first element is the intercept.

Figure 1 displays three observation patterns. In Case 1 ($\delta_i = 1, I_i = 1$), the failure event is observed. In Case 2 ($\delta_i = 0, I_i = 1$), the failure event is not observed due to censoring. In Case 3 ($\delta_i = 0, I_i = 0$) the failure event is not observed due to cure. One cannot distinguish between Case 2 and Case 3 from observation as the value I_i is missing.

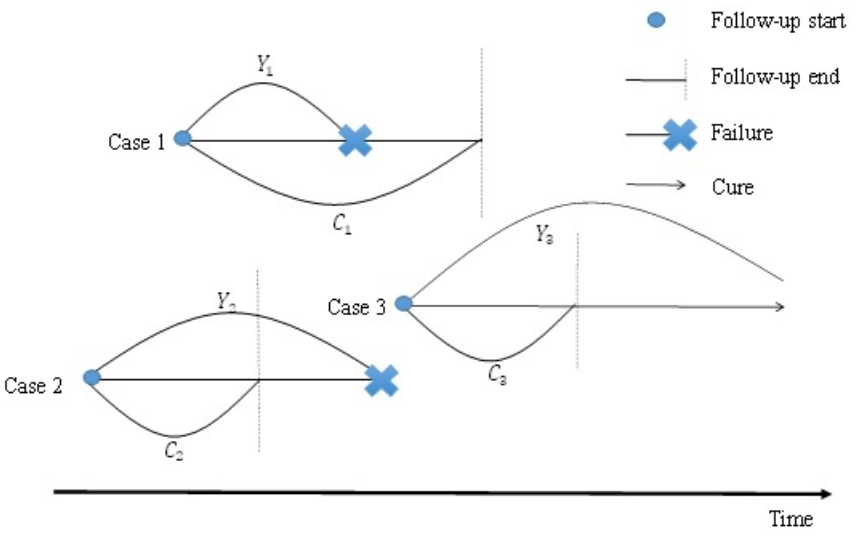

Figure 1: Three observation patterns under censoring and cure.

3.2 Likelihood construction and MLE

Balakrishnan and Pal (2016) relates the cure proportion p_0 to covariates $\mathbf{x}_i' = (1, x_{i1}, \ldots, x_{ik})$ by the logistic link function

$$p_0(\mathbf{x}_i; \boldsymbol{\beta}) = \frac{1}{1 + \exp(\mathbf{x}_i'\boldsymbol{\beta})}, i = 1, 2, \ldots, n,$$

where $\boldsymbol{\beta} = (\beta_0, \beta_1, \ldots, \beta_k)'$ denote the vector of regression coefficients. In the Bernoulli cure case of $\phi \to \infty$, one has $Z(\eta, \phi) = 1 + \eta$, and hence $\eta = \exp(\mathbf{x}_i'\boldsymbol{\beta})$.

Based on the observed data, Rodrigues et al. (2009) obtained the likelihood function

$$L(\boldsymbol{\theta}) = \prod_{i=1}^{n} f_p(t_i, \mathbf{x}_i; \boldsymbol{\theta})^{\delta_i} S_p(t_i, \mathbf{x}_i; \boldsymbol{\theta})^{1-\delta_i},$$

where $\boldsymbol{\theta} = (\boldsymbol{\beta}, \boldsymbol{\gamma})$, $\boldsymbol{t} = (t_1, \ldots, t_n)$, $\boldsymbol{\delta} = (\delta_1, \ldots, \delta_n)$, and $\boldsymbol{\gamma}$ is the parameter related to the distribution of Y. The log-likelihood function is

$$\ell(\boldsymbol{\theta}) = \sum_{i=1}^{n} \delta_i \log f_p(t_i, \mathbf{x}_i; \boldsymbol{\theta}) + \sum_{i=1}^{n} (1 - \delta_i) \log S_p(t_i, \mathbf{x}_i; \boldsymbol{\theta}).$$

The MLE is then defined as $\hat{\boldsymbol{\theta}} = \arg\max_{\theta} \ell(\boldsymbol{\theta})$.

Under the Bernoulli case of $\phi \to \infty$, we have $Z(\eta, \phi) = 1 + \eta$, $\eta = \exp(\mathbf{x_i'}\boldsymbol{\beta})$. Thus, the log-likelihood function can be computed by the expressions

$$S_p(y, \mathbf{x}_i; \boldsymbol{\theta}) = \frac{1 + \exp(\mathbf{x}_i'\boldsymbol{\beta}) S(y; \boldsymbol{\gamma})}{1 + \exp(\mathbf{x}_i'\boldsymbol{\beta})}, \quad f_p(y, \mathbf{x}_i; \boldsymbol{\theta}) = \frac{\exp(\mathbf{x}_i'\boldsymbol{\beta}) f(y; \boldsymbol{\gamma})}{1 + \exp(\mathbf{x}_i'\boldsymbol{\beta})}$$

Example 3. Weibull lifetime with Bernoulli cure

Let $n_1 = \sum_{i=1}^{n} \delta_i$ be the number of uncensored subjects. The log-likelihood function becomes

$$\begin{aligned} \ell(\boldsymbol{\theta}) = &\sum_{i=1}^{n} \delta_i \mathbf{x}_i'\boldsymbol{\beta} - n_1 \log \gamma_1 - \sum_{i=1}^{n} \delta_i \log t_i + \frac{n_1 \log \gamma_2}{\gamma_1} + \frac{1}{\gamma_1} \sum_{i=1}^{n} \delta_i \log t_i - \sum_{i=1}^{n} \delta_i (\gamma_2 t_i)^{\frac{1}{\gamma_i}} \\ &+ \sum_{i=1}^{n} (1 - \delta_i) \log[1 + \exp\{\mathbf{x}_i'\boldsymbol{\beta} - (\gamma_2 t_i)^{\frac{1}{\gamma_1}}\}] - \sum_{i=1}^{n} \log\left\{1 + \exp(\mathbf{x}_i'\boldsymbol{\beta})\right\}, \end{aligned} \tag{1}$$

where $\boldsymbol{\theta} = (\boldsymbol{\beta}, \gamma_1, \gamma_2)$.

Example 4. G-gamma lifetime with Bernoulli cure

We assume that q is a fixed value in $\gamma = (q, \sigma, \lambda)$. Then, the log-likelihood becomes

$$\begin{aligned}\ell(\boldsymbol{\theta}) = &\sum_{i=1}^{n} \delta_i \mathbf{x}_i'\boldsymbol{\beta} + \sum_{i=1}^{n} \delta_i \log f(t_i; \sigma, \lambda) \\ &+ \sum_{i=1}^{n} (1-\delta_i) \log\{1 + \exp(\mathbf{x}_i'\boldsymbol{\beta}) S(t_i; \sigma, \lambda)\} - \sum_{i=1}^{n} \log\{1 + \log(\mathbf{x}_i'\boldsymbol{\beta})\},\end{aligned} \tag{2}$$

where $\boldsymbol{\theta} = (\boldsymbol{\beta}, \sigma, \lambda)$.

3.3 Complete-data likelihood and EM-algorithm

Balakrishnan and Pal (2015, 2016) re-expressed the likelihood function as

$$L(\boldsymbol{\theta}) = \prod_{i=1}^{n} \{f_p(t_i, \mathbf{x}_i; \boldsymbol{\theta})\}^{\delta_i} \prod_{i=1}^{n} \{p_0(\mathbf{x}_i; \boldsymbol{\beta}) + (1 - p_0(\mathbf{x}_i; \boldsymbol{\beta})) S_1(t_i, \mathbf{x}_i; \boldsymbol{\theta})\}^{1-\delta_i}.$$

Assuming that I_i is observable, they considered the complete data likelihood function

$$L_c(\boldsymbol{\theta}) = \prod_{i=1}^{n} \{f_p(t_i, \mathbf{x}_i; \boldsymbol{\theta})^{\delta_i}\}^{I_i} \prod_{i=1}^{n} \{p_0(\boldsymbol{\theta}_1, \mathbf{x}_i)^{1-\delta_i}\}^{1-I_i} \{(1-p_0(\boldsymbol{\theta}_1, \mathbf{x}_i))^{1-\delta_i} S_1(t_i, \mathbf{x}_i; \boldsymbol{\theta})^{1-\delta_i}\}^{I_i}.$$

The log of the complete-data likelihood function is given by

$$\begin{aligned}\ell_c(\boldsymbol{\theta}) = &\sum_{i=1}^{n} I_i \delta_i \log f_p(t_i, \mathbf{x}_i; \boldsymbol{\theta}) + \sum_{i=1}^{n} (1-I_i)(1-\delta_i) \log p_0(\boldsymbol{\theta}_1, \mathbf{x}_i) \\ &+ \sum_{i=1}^{n} I_i (1-\delta_i) \log\{(1 - p_0(\boldsymbol{\theta}_1, \mathbf{x}_i)) S_1(t_i, \mathbf{x}_i; \boldsymbol{\theta})\}.\end{aligned}$$

The idea of the EM algorithm (McLachlan and Krishman 2008) is to remove missing values (I_i's) by taking their conditional expectation given observed data $O = \{$ observed $I_i's$, $(t_i, \delta_i), i = 1, 2, \ldots, n\}$. Given observed data, the $I_i's$ follow the independent Bernoulli random variables with success rate $\pi_i^* = E(I_i|\boldsymbol{\theta}^*, \boldsymbol{O})$, $i = 1, 2, \ldots, n$, where $\boldsymbol{\theta}^* = (\boldsymbol{\beta}^{*'}, \boldsymbol{\gamma}^{*'})'$ are the initial guess for parameters. It follows that

$$\pi_i^* = P_{\theta^*}[I_i = 1 | T_i > t_i] = \frac{P_{\theta^*}[T_i > t_i | I_i = 1] P_{\theta^*}[I_i = 1]}{P_{\theta^*}[T_i > t_i]} = \frac{(1 - p_0(\boldsymbol{\beta}^*, \mathbf{x}_i))\, S_1(t_i, \mathbf{x}_i; \boldsymbol{\theta}^*)}{S_p(t_i, \mathbf{x}_i; \boldsymbol{\theta}^*)}.$$

If $\delta_i = 1$, we set $\pi_i^* = I_i = 1$.

The EM-algorithm of Balakrishnan and Pal (2015, 2016) is stated as follows.

E-Step: Given $\boldsymbol{\theta}^*$, calculate the conditional expectation of the complete-data log-likelihood $Q(\boldsymbol{\theta}, \boldsymbol{\pi}^*) = E[\ell_c(\boldsymbol{\theta})|\boldsymbol{\theta}^*, \boldsymbol{O}]$, where $\boldsymbol{\pi}^*$ is the vector of π_i^*'s.

M-Step: Obtain the updated value $\boldsymbol{\theta}^{**} = \arg\max_{\theta\in\Theta} Q(\boldsymbol{\theta}, \boldsymbol{\pi}^*)$.
The EM algorithm repeats between E-step and M-step until $\boldsymbol{\theta}^{**} \approx \boldsymbol{\theta}^*$. More details are given in Balakrishnan and Pal (2016) for the Weibull model and Balakrishnan and Pal (2015) for the G-gamma model.

It is important to notice that M-Step requires an iteration algorithm to maximize $Q(\boldsymbol{\theta}, \boldsymbol{\pi}^*)$. Balakrishnan and Pal (2015, 2016) used the one-step Newton-Raphson method to approximate $\boldsymbol{\theta}^{**}$. This approximation method, known as the EM-gradient algorithm (Lange 1995), does not strictly maximize $Q(\boldsymbol{\theta}, \boldsymbol{\pi}^*)$. Since it is not clear how this approximation affects the final estimate, we use a strict maximization algorithm in our numerical studies.

4. Newton-Raphson algorithm

This section introduces a robust version of the NR algorithm to find the MLE under the Weibull and G-gamma models. Given the initial values $\boldsymbol{\theta}^*$, the NR algorithm iteration is defined as

$$\boldsymbol{\theta}^{**} = \boldsymbol{\theta}^* - \left\{ \frac{\partial^2}{\partial\boldsymbol{\theta}\partial\boldsymbol{\theta}'}\ell(\boldsymbol{\theta}) \right\}^{-1} \frac{\partial}{\partial\boldsymbol{\theta}}\ell(\boldsymbol{\theta})\Big|_{\theta=\theta^*}.$$

This iteration continues until $\boldsymbol{\theta}^{**} \approx \boldsymbol{\theta}^*$. Under the Bernoulli cure model, we derive the expressions of the first- and second-derivatives in Appendix A (Weibull) and Appendix B (G-gamma).

An important advantage of the NR algorithm over the algorithm of Rodrigues et al. (2009) is the use of the Hessian matrix $\partial^2\ell(\boldsymbol{\theta})/\partial\boldsymbol{\theta}\partial\boldsymbol{\theta}'$. The Hessian matrix not only accelerates the convergence speed, but also facilitates the calculations of standard errors (SEs). The SE of a parameter estimate is a diagonal element of $\left[-\partial^2\ell(\boldsymbol{\theta})/\partial\boldsymbol{\theta}\partial\boldsymbol{\theta}'\big|_{\theta=\hat{\theta}}\right]^{-1}$ that is immediately available from the converged step of the NR algorithm.

A potential disadvantage of the NR algorithm is its sensitivity to the choice of initial

values (Knight 2000). However, this drawback can be easily remedied by adopting two strategies:

- One must use the transformation to avoid the parameter constraints (MacDonald 2014). For instance, to ensure the constraint $\gamma_1 > 0$, the NR algorithm should be performed for the unconstraint parameter $\widetilde{\gamma}_1 = \log(\gamma_1)$.
- One should try the NR algorithm under several different choices of initial values. For instance, the NR algorithm can be tried with *random* initial values. This approach is called the randomized NR algorithm (Hu and Emura 2015; Emura and Pan 2017).

In our experiences, a claim "Newton-Raphson fails to converge" is mostly avoided by the above remedies. Of course, if the likelihood itself has a problem (e.g., does not have a peak), this problem is nothing to do with the NR algorithm. It is most interesting to read the letter of MacDonald (2014) and its reply (Meng 2014) on the arguments between the NR and EM algorithms.

In the following, we provide the concrete algorithms of implementing the parameter transformations and the randomized NR algorithm under the Weibull model.

Algorithm 1: The randomized NR algorithm under the Weibull model

Let $\widetilde{\gamma}_j = \log(\gamma_j)$, $j = 1, 2$, be transformed parameters.

Step 1: Set initial values $(\beta_0^{(0)}, \beta_1^{(0)}, \widetilde{\gamma}_1^{(0)}, \widetilde{\gamma}_2^{(0)})$. We also set some positive tuning parameters B_0 and B_1 , corresponding to β_0 and β_1 .

Step 2: Repeat the following iteration, for $h = 0, 1, 2, \ldots$

$$\begin{bmatrix} \beta_0^{(h+1)} \\ \beta_1^{(h+1)} \\ \widetilde{\gamma}_1^{(h+1)} \\ \widetilde{\gamma}_2^{(h+1)} \end{bmatrix} = \begin{bmatrix} \beta_0^{(h)} \\ \beta_1^{(h)} \\ \widetilde{\gamma}_1^{(h)} \\ \widetilde{\gamma}_2^{(h)} \end{bmatrix} - \begin{bmatrix} \frac{\partial^2 \ell}{\partial \beta_0^2} & \frac{\partial^2 \ell}{\partial \beta_0 \partial \beta_1} & \frac{\partial^2 \ell}{\partial \beta_0 \partial \gamma_1} & \frac{\partial^2 \ell}{\partial \beta_0 \partial \beta_2} \\ \frac{\partial^2 \ell}{\partial \beta_0 \partial \beta_1} & \frac{\partial^2 \ell}{\partial \beta_1^2} & \frac{\partial^2 \ell}{\partial \beta_1 \partial \gamma_1} & \frac{\partial^2 \ell}{\partial \beta_1 \partial \gamma_2} \\ \frac{\partial^2 \ell}{\partial \beta_0 \partial \gamma_1} & \frac{\partial^2 \ell}{\partial \beta_1 \partial \gamma_1} & \frac{\partial^2 \ell}{\partial \gamma_1^2} & \frac{\partial^2 \ell}{\partial \gamma_1 \partial \gamma_2} \\ \frac{\partial^2 \ell}{\partial \beta_0 \partial \gamma_2} & \frac{\partial^2 \ell}{\partial \beta_1 \partial \gamma_2} & \frac{\partial^2 \ell}{\partial \gamma_1 \partial \gamma_2} & \frac{\partial^2 \ell}{\partial \gamma_2^2} \end{bmatrix}^{-1} \begin{bmatrix} \frac{\partial \ell}{\partial \beta_0} \\ \frac{\partial \ell}{\partial \beta_1} \\ \frac{\partial \ell}{\partial \gamma_1} \\ \frac{\partial \ell}{\partial \gamma_2} \end{bmatrix} \Bigg|_{\substack{\beta_0 = \beta_0^{(h)}, \beta_1 = \beta_1^{(h)}, \\ \gamma_1 = \exp(\widetilde{\gamma}_1^{(h)}), \\ \gamma_2 = \exp(\widetilde{\gamma}_2^{(h)})}}$$

- if $\max\{|\beta_0^{(h+1)} - \beta_0^{(h)}|, |\beta_1^{(h+1)} - \beta_1^{(h)}|, |\widetilde{\gamma}_1^{(h+1)} - \widetilde{\gamma}_1^{(h)}|, |\widetilde{\gamma}_2^{(h+1)} - \widetilde{\gamma}_2^{(h)}|\} < 10^{-3}$, then stop the algorithm and $(\beta_0^{(h+1)}, \beta_1^{(h+1)}, \widetilde{\gamma}_1^{(h+1)}, \widetilde{\gamma}_2^{(h+1)})$ is the MLE of $(\beta_0, \beta_1, \widetilde{\gamma}_1, \widetilde{\gamma}_2)$.
- if $|\beta_0^{(h+1)}| > B_0$, replace $\beta_0^{(0)}$ with $\beta_0^{(0)} + u_1$, where $u_1 \sim U(-1, 1)$. Return to Step 2.
- if $|\beta_1^{(h+1)}| > B_1$, replace $\beta_1^{(0)}$ with $\beta_1^{(0)} + u_2$, where $u_2 \sim U(-1, 1)$. Return to Step 2.

Appendix C provides the R codes for implementing Algorithm 1 in the simulation studies. In the codes, we set initial values $\beta_0^{(0)} = \beta_1^{(0)} = 0$ and $\widetilde{\gamma}_1^{(0)} = \widetilde{\gamma}_2^{(0)} = -2$. The choice "$-2$" means that initial values $\gamma_1^{(0)}$ and $\gamma_2^{(0)}$ are near zero. However, the choice "-3" is too close to zero that yields a Hessian matrix that is singular (not invertible). This phenomenon yields a condition that the parameter space for γ_j is (ε, ∞) for a small value $\varepsilon > 0$. The tuning parameters in Step 1 are chosen to be $B_0 = B_1 = 3$ in the simulations and data analysis. However, the choice is somewhat arbitrary and depends on the scale of β_0, $\beta_1, \widetilde{\gamma}_1$, and $\widetilde{\gamma}_2$. Therefore, we examine the robustness of $B_0 = B_1 = 3$ under different parameter settings in the simulations.

5. Simulation

We conducted Monte Carlo simulations to compare the performance between the NR and EM algorithms. We adopt the same design as Balakrishnan and Pal (2015, 2016) that used a discrete covariate (x = 1, 2, 3, or 4). We focus our simulations on the Bernoulli cure model with the Weibull or the G-gamma models. Model parameters were chosen to control the cure proportion and censoring proportion (heavy vs. moderate) at each x. Table 2 summarizes the parameter configuration.

We generated data by following the schemes of Balakrishnan and Pal (2015, 2016), and then calculated the parameter estimates by using the NR and EM algorithms. For each algorithm, we counted the number of iterations to achieve convergence. Note that

Table 2: Parameter configurations for simulations.

i) Weibull model: $\beta_0 = -1.192$, $\beta_1 = 0.573$, $\gamma_1 = 0.316$, $\gamma_2 = 0.179$				
Covariate x	1	2	3	4
Cure proportion $p_0(x;\beta_0,\beta_1)$	0.65	0.51	0.37	0.25
Heavy censoring proportion p	0.80	0.65	0.50	0.35
ii) Weibull model: $\beta_0 = -0.038$, $\beta_1 = 0.443$, $\gamma_1 = 0.316$, $\gamma_2 = 0.179$				
Covariate x	1	2	3	4
Cure proportion $p_0(x;\beta_0,\beta_1)$	0.40	0.30	0.22	0.15
Moderate censoring proportion	0.50	0.40	0.30	0.20
iii) G-gamma model: $\beta_0 = -1.192$, $\beta_1 = 0.573$, $\sigma = 0.871$, $\lambda = 4.632$, $q = 0.5$				
Covariate x	1	2	3	4
Cure proportion $p_0(x;\beta_0,\beta_1)$	0.65	0.51	0.37	0.25
Heavy censoring proportion	0.80	0.65	0.50	0.35
iv) G-gamma model: $\beta_0 = -0.038$, $\beta_1 = 0.443$, $\sigma = 0.871$, $\lambda = 4.632$, $q = 0.5$				
Covariate x	1	2	3	4
Cure proportion $p_0(x;\beta_0,\beta_1)$	0.40	0.30	0.22	0.15
Moderate censoring proportion	0.50	0.40	0.30	0.20

the EM algorithm has two different counts of iterations, "Outside iteration" between E-step and M- steps and "Inside iteration" within M-step. We also calculated the SE and the 95% confidence interval (CI) to see the performance of interval estimation. The estimates and coverage rates (CRs) of the CIs are assessed based on 200 repetitions.

The R codes for the simulation studies are available in Appendix C.

Results under the Weibull model

Results are given in Table 3 (NR) and Table 4 (EM).

The NR and EM algorithms produced nearly identical estimation performances (Tables 3 and 4). The NR and EM gave unbiased estimates for the true values for all the configurations, irrespective of censoring percentages and cure proportions. Both algorithms achieved the CRs close to the nominal levels in most configurations. One remarkable exception is the CR for γ_2 which is slightly below the nominal level. The

NR algorithm has especially worth performance (coverage rate=0.845 for the nominal 90%), which is somehow improved by the EM (coverage rate=0.870). While the reason is not clear, we used the log-transformed CI intervals for both γ_1 and γ_2 . Perhaps, other transformations may improve the coverage performance since SE is still close to SD.

The main difference between the NR and EM algorithms is the number of iterations until convergence. The NR requires about 30 iterations. This represents the total number of iterations including the steps of random initial values. The EM algorithm requires about 5 outside iterations (between E- and M- steps) and 8 inside iterations within M-step, thus, 5×8=40 iterations in total. Hence, there seems no big difference in their actual iteration numbers and thus their computing times.

Recall that the NR algorithm (Algorithm 1) employs the tuning parameters B_0 and B_1. For instance, if $|\beta_0^{(h+1)}| > B_0$ in the h-th iteration, the NR algorithm restarts after replacing the initial values by $\beta_0^{(0)} + u_1$, where $u_1 \sim U(-1, 1)$. We shall examine the robustness of our chosen values $B_0 = B_1 = 3$ under different parameter settings than those of Tables 3 and Tables 4. Table 5 shows the simulation results by replacing $\gamma_1 = 0.316$ and $\gamma_2 = 0.179$ with $\gamma_1 = \gamma_2 = 1.2$. Estimates are still unbiased and the CIs have accurate CRs for the true parameters.

Results under the G-gamma model

Results are given in Table 6 (NR) and Table 7 (EM).

The NR and EM algorithms produced similar estimation performances (Estimate, SD, SE are very similar but CR is not). The NR and EM algorithms gave unbiased estimates for the true values for all the configurations, irrespective of censoring percentages and cure proportions. Both algorithms achieved the CRs reasonably close to the nominal levels in most configurations. An exception is the CR for γ_2 , where the worst case is seen in both the NR and EM algorithms (CR=0.845 for the nominal 90% in n=400). However, this is improved by increasing the sample size (CR=0.890 for the nominal 90% in n=600).

The main difference between the NR and EM algorithms is the number of iterations

Table 3: Simulation results on the NR algorithm with Weibull lifetime (200 repetitions).

n	Censoring proportion	AI	Parameter	Estimate	Bias	SD	SE	CR	
								90%	95%
200	High	30.5	β_0 =-1.192	-1.236	-0.044	0.411	0.393	0.895	0.960
			$\beta_1 = 0.573$	0.584	0.011	0.163	0.156	0.900	0.945
			$\gamma_1 = 0.316$	0.314	-0.002	0.025	0.025	0.895	0.940
			$\gamma_2 = 0.179$	0.179	0.000	0.007	0.006	0.870	0.935
	Low	30.6	β_0 =-0.038	-0.038	0.000	0.401	0.394	0.905	0.950
			$\beta_1 = 0.443$	0.446	0.003	0.167	0.167	0.915	0.945
			$\gamma_1 = 0.316$	0.314	-0.002	0.021	0.021	0.885	0.935
			$\gamma_2 = 0.179$	0.178	-0.001	0.006	0.005	0.845	0.910
400	High	30.5	β_0 =-1.192	-1.236	-0.044	0.266	0.277	0.915	0.955
			$\beta_1 = 0.573$	0.588	0.015	0.107	0.110	0.900	0.965
			$\gamma_1 = 0.316$	0.314	-0.002	0.017	0.018	0.915	0.950
			$\gamma_2 = 0.179$	0.179	0.000	0.005	0.004	0.860	0.910
	Low	30.6	β_0 = -0.038	-0.044	-0.006	0.281	0.278	0.890	0.945
			$\beta_1 = 0.443$	0.448	0.005	0.120	0.117	0.900	0.945
			$\gamma_1 = 0.316$	0.314	-0.002	0.016	0.015	0.900	0.930
			$\gamma_2 = 0.179$	0.179	-0.000	0.004	0.004	0.845	0.925
600	High	30.5	β_0 =-1.192	-1.197	-0.005	0.226	0.225	0.900	0.960
			$\beta_1 = 0.573$	0.574	0.001	0.089	0.089	0.885	0.945
			$\gamma_1 = 0.316$	0.315	-0.001	0.014	0.015	0.925	0.970
			$\gamma_2 = 0.179$	0.179	0.000	0.004	0.004	0.905	0.950
	Low	30.6	β_0 =-0.038	-0.055	-0.017	0.241	0.227	0.910	0.955
			$\beta_1 = 0.443$	0.458	0.015	0.101	0.096	0.870	0.920
			$\gamma_1 = 0.316$	0.315	-0.001	0.013	0.012	0.885	0.955
			$\gamma_2 = 0.179$	0.179	0.000	0.003	0.003	0.870	0.940

Note: AI=the average number of iterations, Estimate=the average of estimates, SD=standard deviation, SE=the average of standard errors, CR=coverage rate of confidence intervals.

Table 4: Simulation results on the EM algorithm with Weibull lifetime (200 repetitions).

n	p	AI		Parameter	Estimate	Bias	SD	SE	CR	
		outside	inside						90%	95%
200	High	5.58	7.6	β_0 =-1.192	-1.236	-0.044	0.411	0.393	0.895	0.960
				$\beta_1 = 0.573$	0.584	0.011	0.163	0.156	0.900	0.945
				$\gamma_1 = 0.316$	0.314	-0.002	0.025	0.025	0.905	0.940
				$\gamma_2 = 0.179$	0.179	0.000	0.007	0.006	0.870	0.940
	Low	4.3	8.7	β_0 =-0.038	-0.037	-0.001	0.401	0.394	0.905	0.950
				$\beta_1 = 0.443$	0.446	0.003	0.167	0.167	0.915	0.945
				$\gamma_1 = 0.316$	0.315	-0.001	0.021	0.021	0.890	0.935
				$\gamma_2 = 0.179$	0.179	0.000	0.006	0.005	0.880	0.930
400	High	5.6	7.6	β_0 =-1.192	-1.235	-0.043	0.266	0.277	0.915	0.955
				$\beta_1 = 0.573$	0.588	0.015	0.107	0.110	0.900	0.965
				$\gamma_1 = 0.316$	0.315	-0.001	0.017	0.018	0.915	0.955
				$\gamma_2 = 0.179$	0.179	0.000	0.005	0.004	0.870	0.920
	Low	4.2	8.8	β_0 = -0.038	-0.044	-0.006	0.281	0.278	0.890	0.945
				$\beta_1 = 0.443$	0.448	0.005	0.120	0.117	0.900	0.945
				$\gamma_1 = 0.316$	0.315	-0.001	0.016	0.015	0.915	0.935
				$\gamma_2 = 0.179$	0.179	0.000	0.004	0.004	0.880	0.935
600	High	5.6	7.6	β_0 =-1.192	-1.197	-0.005	0.226	0.225	0.900	0.960
				$\beta_1 = 0.573$	0.574	0.001	0.089	0.089	0.885	0.945
				$\gamma_1 = 0.316$	0.316	-0.000	0.014	0.015	0.925	0.970
				$\gamma_2 = 0.179$	0.179	0.000	0.004	0.004	0.915	0.950
	Low	4.1	9.0	β_0 =-0.038	-0.055	-0.017	0.241	0.227	0.910	0.955
				$\beta_1 = 0.443$	0.458	0.015	0.101	0.096	0.870	0.920
				$\gamma_1 = 0.316$	0.316	-0.000	0.012	0.012	0.890	0.955
				$\gamma_2 = 0.179$	0.179	0.000	0.003	0.003	0.870	0.935

Note: p =censoring proportion, Estimate=the average of estimates, AI=the average number of iterations (Outside for E-steps, Inside for Q-function), SD=standard deviation, SE=the average of standard errors, CR=coverage rate of confidence intervals.

Table 5: Simulation results on the NR algorithm with Weibull lifetime (200 repetitions).

n	Censoring proportion	AI	Parameter	Estimate	Bias	SD	SE	CR	
								90%	95%
200	High	22.9	β_0 =-1.192	-1.225	-0.033	0.381	0.368	0.910	0.955
			$\beta_1 = 0.573$	0.580	0.007	0.150	0.147	0.910	0.965
			$\gamma_1 = 1.2$	1.193	-0.007	0.091	0.093	0.910	0.955
			$\gamma_2 = 1.2$	1.218	0.018	0.165	0.153	0.880	0.940
	Low	22.7	β_0 =-0.038	-0.033	0.005	0.366	0.378	0.935	0.955
			$\beta_1 = 0.443$	0.445	0.002	0.152	0.161	0.915	0.965
			$\gamma_1 = 1.2$	1.195	-0.005	0.075	0.079	0.910	0.940
			$\gamma_2 = 1.2$	1.212	0.012	0.142	0.129	0.870	0.930
400	High	22.5	β_0 =-1.192	-1.222	-0.030	0.252	0.259	0.920	0.960
			$\beta_1 = 0.573$	0.582	0.009	0.102	0.103	0.915	0.965
			γ_1 =1.2	1.194	-0.006	0.062	0.066	0.915	0.960
			$\gamma_2 = 1.2$	1.215	0.015	0.120	0.108	0.855	0.915
	Low	22.5	β_0 = -0.038	-0.035	-0.003	0.263	0.266	0.890	0.945
			$\beta_1 = 0.443$	0.444	0.001	0.114	0.113	0.875	0.955
			$\gamma_1 = 1.2$	1.196	-0.004	0.057	0.056	0.895	0.945
			$\gamma_2 = 1.2$	1.215	0.015	0.098	0.092	0.860	0.920
600	High	22.4	β_0 =-1.192	-1.193	-0.001	0.213	0.211	0.910	0.950
			$\beta_1 = 0.573$	0.572	-0.001	0.085	0.084	0.890	0.940
			$\gamma_1 = 1.2$	1.199	-0.001	0.050	0.054	0.925	0.960
			$\gamma_2 = 1.2$	1.213	0.013	0.086	0.088	0.900	0.945
	Low	22.4	β_0 =-0.038	-0.050	-0.012	0.229	0.218	0.895	0.940
			$\beta_1 = 0.443$	0.456	0.013	0.095	0.093	0.885	0.955
			$\gamma_1 = 1.2$	1.199	-0.001	0.046	0.046	0.895	0.945
			$\gamma_2 = 1.2$	1.214	0.014	0.080	0.075	0.860	0.935

Note: AI=the average number of iterations, Estimate=the average of estimates, SD=standard deviation, SE=the average of standard errors, CR=coverage rate of confidence intervals.

Table 6: Simulation results on the NR algorithm with the G-gamma model (200 repetitions).

n	Censoring proportion	AI	Parameter	Estimate	Bias	SD	SE	CR	
								90%	95%
200	High	13.8	β_0 =-1.192	-1.231	-0.039	0.107	0.399	0.925	0.965
			β_1 = 0.573	0.585	0.012	0.164	0.159	0.910	0.960
			σ = 0.871	0.869	-0.002	0.072	0.070	0.870	0.945
			λ =4.632	4.635	0.003	0.461	0.447	0.885	0.950
	Low	14.0	β_0 =-0.038	-0.049	-0.011	0.424	0.403	0.895	0.945
			β_1 = 0.443	0.455	0.011	0.174	0.171	0.910	0.950
			σ = 0.871	0.869	-0.002	0.053	0.057	0.915	0.960
			λ = 4.623	4.674	0.015	0.399	0.366	0.865	0.940
400	High	13.8	β_0 =-1.192	-1.221	-0.029	0.274	0.280	0.915	0.960
			β_1 = 0.573	0.583	0.010	0.112	0.112	0.890	0.945
			σ = 0.871	0.865	-0.006	0.047	0.049	0.890	0.960
			λ = 4.632	4.635	0.003	0.353	0.314	0.870	0.925
	Low	13.5	β_0 = -0.038	-0.046	-0.008	0.291	0.283	0.905	0.940
			β_1 = 0.443	0.451	0.008	0.126	0.120	0.865	0.920
			σ = 0.871	0.866	-0.005	0.039	0.040	0.915	0.935
			λ = 4.632	4.622	-0.010	0.285	0.256	0.845	0.905
600	High	13.6	β_0 =-1.192	-1.188	0.004	0.230	0.228	0.905	0.950
			β_1 = 0.573	0.570	-0.003	0.091	0.091	0.895	0.945
			σ = 0.871	0.869	-0.002	0.037	0.040	0.910	0.960
			λ = 4.632	4.612	-0.020	0.261	0.256	0.920	0.950
	Low	14.1	β_0 =-0.038	-0.051	-0.013	0.247	0.231	0.880	0.955
			β_1 = 0.443	0.456	0.013	0.103	0.098	0.890	0.945
			σ = 0.871	0.869	-0.002	0.032	0.033	0.900	0.945
			λ = 4.632	4.618	-0.014	0.228	0.209	0.880	0.930

Note: AI=the average number of iterations, Estimate=the average of estimates, SD=standard deviation, SE=the average of standard errors, CR=coverage rate of confidence intervals.

Table 7: Simulation results on the EM with the G-gamma model (200 repetitions).

n	p	AI		Parameter	Estimate	Bias	SD	SE	CR	
		outside	inside						90%	95%
200	High	6.2	13.5	β_0 =-1.192	-1.231	-0.039	0.407	0.399	0.925	0.965
				$\beta_1 = 0.573$	0.585	0.012	0.164	0.159	0.910	0.960
				$\sigma = 0.871$	0.869	-0.002	0.072	0.070	0.880	0.945
				$\lambda = 4.632$	4.635	0.003	0.460	0.447	0.890	0.940
	Low	4.9	14.7	β_0 = -0.038	-0.049	-0.011	0.424	0.403	0.895	0.945
				$\beta_1 = 0.443$	0.454	0.011	0.174	0.171	0.910	0.950
				$\sigma = 0.871$	0.869	-0.002	0.053	0.057	0.910	0.960
				$\lambda = 4.632$	4.648	0.016	0.399	0.366	0.865	0.940
400	High	6.1	13.8	β_0 =-1.192	-1.221	-0.029	0.274	0.280	0.915	0.960
				$\beta_1 = 0.573$	0.584	0.011	0.112	0.112	0.890	0.945
				$\sigma = 0.871$	0.865	-0.006	0.047	0.0494	0.890	0.960
				$\lambda = 4.632$	4.635	0.003	0.353	0.314	0.870	0.925
	Low	4.7	14.5	β_0 = -0.038	-0.046	-0.008	0.291	0.283	0.905	0.940
				$\beta_1 = 0.443$	0.451	0.008	0.126	0.120	0.865	0.920
				$\sigma = 0.871$	0.866	-0.005	0.039	0.004	0.915	0.935
				$\lambda = 4.632$	4.622	-0.010	0.285	0.256	0.845	0.905
600	High	6.1	13.7	β_0 =-1.192	-1.188	0.004	0.230	0.228	0.905	0.955
				$\beta_1 = 0.573$	0.570	-0.003	0.091	0.091	0.895	0.945
				$\sigma = 0.871$	0.869	-0.002	0.037	0.040	0.910	0.960
				$\lambda = 4.632$	4.611	-0.021	0.260	0.256	0.920	0.950
	Low	4.6	14.6	β_0 =-0.038	-0.051	-0.013	0.247	0.231	0.880	0.955
				$\beta_1 = 0.443$	0.456	0.013	0.102	0.098	0.885	0.945
				$\sigma = 0.871$	0.869	-0.002	0.032	0.033	0.900	0.945
				$\lambda = 4.632$	4.618	-0.014	0.228	0.209	0.890	0.925

Note: p=censoring proportion, AI=the average number of iterations (Outside for E-steps, Inside for Q-function), Estimate=the average of estimates, SD=standard deviation, SE=the average of standard errors, CR=coverage rate of confidence intervals.

to achieve convergence. The NR algorithm requires about 14 iterations to achieve convergence. The EM algorithm also requires 14 inside iterations within M-step. Since the EM requires about 5 outside iterations between E- and M-steps, the actual iteration number is 5×14=70. Therefore, the NR algorithm converges much faster than the EM algorithm.

6. Data analysis

We consider the uveal cancer data to illustrate the methods.

6.1 Uveal cancer data

We use the uveal cancer data available from European Molecular Biology Laboratory-European Bioinformatics Institute (EMBL-EBI). The data are obtained from EMBL-EBI's website (http://www.ebi.ac.uk/arrayexpress/experiments/E-GEOD-22138/), which were released on December 2010 and updated on March 2012.

Uveal melanoma is a cancer of eyes, occurring in left or right eye. Uveal cancer patients may develop metastasis frequently occurring in the liver. Therefore, time-to-metastasis is the event time of interest.

Table 8 shows the profile of $n = 63$ patients. The event time is time-to-metastasis (in months, denoted as Y). The event time is censored by follow-up time (in months, denoted as C). The gender is a binary covariate of interest. There are 39 males and 24 females.

6.2 Model fitting

In the Bernoulli cure model, the cure proportion is related to a covariate x_i by

$$p_0(x_i;\beta_0,\beta_1) = \frac{1}{1+\exp(\beta_0+\beta_1 x_i)}, i = 1, 2, \ldots, n.$$

where x_i is the gender (set $x_i = 1$ for males) and (set $x_i = 2$ for females). Under this model, the improper survival function is

$$S_p(y, x_i; \boldsymbol{\theta}) = \frac{1+\exp(\beta_0+\beta_1 x_i)S(y;\boldsymbol{\gamma})}{1+\exp(\beta_0+\beta_1 x_i)},$$

where the form of the survival function $S(y;\boldsymbol{\gamma})$ depends on the lifetime model. The estimates of these quantities are obtained by replacing the parameters by their MLEs (calculated by either the EM or the NR algorithm). The SEs are estimated by the delta method, and 95% CIs are computed by the normal approximation under the usual assumption of the asymptotic normality of the parameter estimates.

1) Weibull lifetime model: $S(y;\boldsymbol{\gamma}) = \exp\{-(\gamma_2 y)^{\frac{1}{\gamma_1}}\}$

Table 9 shows the parameter estimates obtained from the NR and EM algorithms. The two algorithms produce similar estimates. Although the numerical values are not identical, the NR and EM algorithms essentially produce the same conclusion: Females have a higher cure rate (38%) than males (30%). This is confirmed by the fitted improper survival curves (Figure 2).

2) The G-gamma lifetime model: $S(y;\boldsymbol{\gamma}) = \Gamma(q^{-2}, q^{-2}(\lambda y)^{\frac{q}{\sigma}})/\Gamma(q^{-2})$

Table 10 shows the estimates and the likelihood values in the range of $q \in [0.5, 2]$ based on the NR algorithm. At the row of $q = 1$ (the Weibull model), estimates are identical to those in Table 9, meaning that the NR algorithm developed under the G-gamma model reduces to its special case. Table 10 shows that the likelihood value is maximized at $q = 1.8$. Hence, we use this value for the subsequent analysis.

Table 11 shows the parameter estimates obtained from the NR and EM algorithms. The two algorithms produce very similar estimates, showing no practical difference. The conclusion from this data analysis is that females have a higher cure rate (42%) than males do (35%). The same conclusion was seen under the Weibull model, but their cure rates are slightly smaller (38% for female and 30% for male). This is the effect of having the flat tails of the gamma-based survival function (Figure 3).

Table 8: The uveal cancer data (http://www.ebi.ac.uk/arrayexpress/experiments /E-GEOD-22138/).

Number	**Source Name**	**Event time (months)**	**Event status (yes=metastasis)**	**Gender**
1	GSM550623 1	73	no	female
2	GSM550624 1	32.13	yes	male
3	GSM550625 1	0.39	yes	male
⋮	⋮	⋮	⋮	⋮
63	GSM550685 1	7.59	yes	male

NOTE: Censored percentage is 46% and the event time has median=32.13 (months) and IQR=50.79 (months). The full data is available in Appendix D of He (2017).

Table 9: Results for fitting the Weibull model to the uveal cancer data.

		Parameter	Estimate	95 %CI
NR		β_0	1.236	(-1.168, 3.641)
		β_1	-0.367	(-1.740, 1.005)
		γ_1	1.002	(0.616, 1.387)
		γ_2	0.028	(0.008, 0.047)
	Cure rate	Male	0.295	(0.059, 0.556)
		Female	0.377	(0.138, 0.634)
EM		β_0	1.162	(-1.073. 3.398)
		β_1	-0.349	(-1.667, 0.969)
		γ_1	0.987	(0.627, 1.348)
		γ_2	0.030	(0.011, 0.049)
	Cure rate	Male	0.307	(0.080, 0.534)
		Female	0.386	(0.159, 0.613)

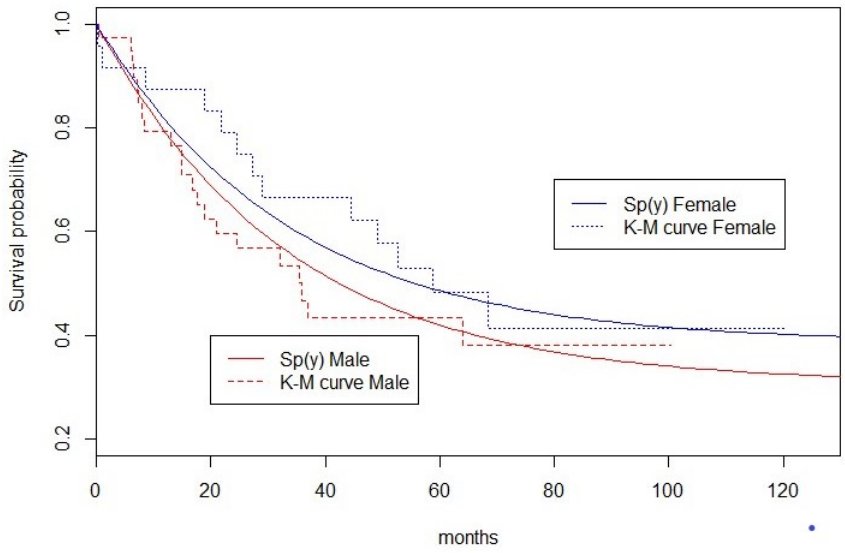

Figure 2: Results for fitting the Weibull model to the uveal cancer data (gender as a covariate). Kaplan-Meier curves and fitted parametric survival curves are shown.

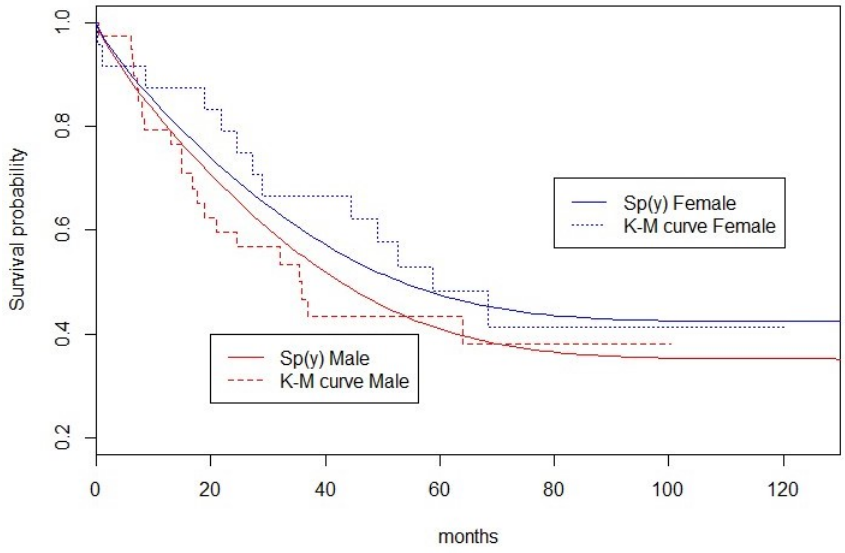

Figure 3: Results for fitting the G-gamma model to the uveal cancer data (gender as a covariate). Kaplan-Meier curves and fitted parametric survival curves are shown.

Table 10: Results for fitting the G-gamma model to the uveal cancer data.

	β_0	β_1	σ	λ	likelihood value
q =0.5	9.345	3.903	1.596	0.015	-179.9056
q =0.6	2.606	-0.724	1.372	0.022	-179.5428
q =0.7	1.932	-0.529	1.252	0.025	-179.3688
q =0.8	1.582	-0.443	1.152	0.027	-179.1908
q =0.9	1.371	-0.396	1.070	0.028	-179.0193
q =1.0 (Weibull)	**1.233**	**-0.367**	**1.001**	**0.029**	**-178.8632**
q =1.1	1.139	-0.347	0.944	0.029	-178.7283
q =1.2	1.073	-0.334	0.894	0.029	-178.6174
q =1.3	1.025	-0.325	0.851	0.029	-178.5306
q =1.4	0.990	-0.318	0.812	0.028	-178.4662
q =1.5	0.963	-0.313	0.777	0.028	-178.4215
q =1.6	0.944	-0.309	0.746	0.027	-178.3934
q =1.7	0.930	-0.307	0.716	0.027	-178.3786
q =1.8 (Maximized)	**0.919**	**-0.306**	**0.689**	**0.026**	**-178.3741**
q =1.9	0.912	-0.305	0.664	0.025	-178.3773
q =2.0	0.907	-0.304	0.640	0.025	-178.3857

Table 11: Results for fitting the G-gamma model ($q = 1.8$) to the uveal cancer data.

		Parameter	Estimate	95 %CI
NR		β_0	0.919	(-0.900, 2.738)
		β_1	-0.306	(-1.471, 0.860)
		γ_1	0.689	(0.469, 0.910)
		γ_2	0.026	(0.017, 0.035)
	Cure rate	Male	0.351	(0.175, 0.527)
		Female	0.424	(0.248, 0.600)
EM		β_0	0.918	(-0.980, 2.815)
		β_1	-0.305	(-1.474, 0.863)
		γ_1	0.689	(0.430, 0.948)
		γ_2	0.026	(0.016, 0.036)
	Cure rate	Male	0.351	(0.149, 0.554)
		Female	0.424	(0.221, 0.626)

7. Conclusion and discussion

The main objective of this paper was to study the computational aspects of the COM-Poisson cure rate models for survival data. We developed the NR algorithm that is a simple, but has not been considered for the COM-Poisson cure models. We proposed a robust version of a NR algorithm, which is robust against the choice of the initial values. We also revisited the EM algorithms of Balakrishnan and Pal (2015, 2016).

According to our numerical studies, the NR and EM algorithms show similar performance in terms of estimates. This is because the two algorithms are intended to maximize the same likelihood. However, we still feel the need to check this agreement due to the remarkable difference in their mathematical derivations. To the best of our knowledge, no paper has yet checked this important issue.

Our conclusion is that the NR algorithm is faster to converge than the EM algorithm, which agrees with the accepted wisdom. This is due to the fact that the EM algorithm requires two different iterations (i.e., inside iterations and outside iterations). However, in the present context, the speed is not significant enough to speak out for the superiority of the NR algorithm. If one uses the EM-gradient algorithm, as suggested in Balakrishnan and Pal (2015, 2016), one can shorten the M-step iterations. However, the EM-gradient algorithm is an approximation technique, so it may not truly maximize the likelihood. It is not clear for us if this approximation technique should be applied only for the purpose to reduce the computational speed.

In the real data analysis, we concluded that the Bernoulli cure model with the G-gamma model is a suitable model for the uveal cancer data. Interestingly, we found that the G-gamma model yielded flatter survival curves at the tails than the Weibull did. In this respect, the G-gamma model developed by Balakrishnan and Pal (2015) is an appealing choice since the primary role of the cure model is to detect "flat" tails in the survival curves.

In the literature, it is assumed that competing event times $W_j (j = 1, 2, \ldots, m)$ are iid with a common distribution $F(y) = Pr(W_j \leq y)$. Since m event types may be associated in many cancer examples, the independence assumption may be questionable. To relax the independence assumption, one may consider the shared frailty model, where

dependence among event times is accounted by an unobserved frailty term (Duchateau and Janssen 2007). Alternatively, dependence among event times is modeled via a copula function (Nelsen 2006). In either case, careful consideration of identifiability (Tsiatis 1975) is required since one can only observe the event time $T = \min(Y, C)$, where $Y = \min\{W_1, W_2, \ldots, W_m\}$.

The independent censoring assumption between Y and C is another questionable assumption to be examined. The independence assumption can often be relaxed by copulas (Emura and Chen 2018). Since the dependence structure between Y and C is usually asymmetric, it is interesting to study the "directional dependence" via asymmetric copulas (Kim et al. 2009). So far, copula models have not been considered under the COM-Poisson cure model and are interesting topics for future developments.

The cancer data we analyzed contain genomic information but we did not use them. Then, it is an important computational issue to develop a method to incorporate high-dimensional genetic information into the model. To reduce the dimension of the genetic data, one can perform univariate feature selection based on significance tests with the Cox models (Emura et al. 2019). Penalized Cox regression methods, such as the ridge regression and Lasso, and tree-based ensemble methods can also be considered (van Wieringen et al. 2009). However, these methods developed for Cox regression should be modified to the logistic regression for the COM-Poisson cure rate model.

Acknowledgement

The authors kindly thank the associate editor and the anonymous reviewer for their valuable suggestions that improved the paper. This research is supported by Ministry of Science and Technology, Taiwan (MOST 103-2118-M-008-MY2).

Appendix A: Derivatives of the log-likelihood under the Weibull model

Recall from Equation (1) that the log-likelihood function is

$$\begin{aligned}\ell(\boldsymbol{\theta};\boldsymbol{t},\boldsymbol{\delta}) = &\sum_{i=1}^{n}\delta_i(\mathbf{x}_i'\boldsymbol{\beta}) - n_1\log(\gamma_1) - \sum_{i=1}^{n}\delta_i\log(t_i) + \frac{n_1\log(\gamma_2)}{\gamma_1} + \frac{1}{\gamma_1}\sum_{i=1}^{n}\delta_i\log(t_i)\\ &- \sum_{i=1}^{n}\delta_i(\gamma_2 t_i)^{\frac{1}{\gamma_1}} + \sum_{i=1}^{n}(1-\delta_i)\log\{1+\exp((\mathbf{x}_i'\boldsymbol{\beta}) - (\gamma_2 t_i)^{\frac{1}{\gamma_1}})\}\\ &- \sum_{i=1}^{n}\log\{1+\exp(\mathbf{x}_i'\boldsymbol{\beta})\}\end{aligned}$$

where $n_1 = \sum_{i=1}^{n}\delta_i$. Let $D_i(\boldsymbol{\theta}) = (\mathbf{x}_i'\boldsymbol{\beta}) - (\gamma_2 t_i)^{\frac{1}{\gamma_1}}$. Then,

$$\frac{\partial D_i(\boldsymbol{\theta})}{\partial\beta_0} = 1, \frac{\partial D_i(\boldsymbol{\theta})}{\partial\beta_1} = x_i, \frac{\partial D_i(\boldsymbol{\theta})}{\partial\gamma_1} = (\gamma_2 t_i)^{\frac{1}{\gamma_1}}\log(\gamma_2 t_i)\frac{1}{\gamma_1^2},$$

$$\frac{\partial D_i(\boldsymbol{\theta})}{\partial\gamma_2} = -\frac{1}{\gamma_1\gamma_2}(\gamma_2 t_i)^{\frac{1}{\gamma_1}}, \frac{\partial^2 D_i(\boldsymbol{\theta})}{\partial\gamma_1^2} = (\gamma_2 t_i)^{\frac{1}{\gamma_1}}\log(\gamma_2 t_i)\frac{1}{\gamma_1^3}\Big\{-\log(\gamma_2 t_i)\frac{1}{\gamma_1} - 2\Big\},$$

$$\frac{\partial^2 D_i(\boldsymbol{\theta})}{\partial\gamma_1\partial\gamma_2} = \frac{1}{\gamma_1^2\gamma_2}(\gamma_2 t_i)^{\frac{1}{\gamma_1}}\Big\{1+\frac{1}{\gamma_1}\log(\gamma_2 t_i)\Big\}, \frac{\partial^2 D_i(\boldsymbol{\theta})}{\partial\gamma_2^2} = \frac{1}{\gamma_1\gamma_2^2}(\gamma_2 t_i)^{\frac{1}{\gamma_1}}\left(1-\frac{1}{\gamma_1}\right).$$

The 1st derivatives:

$$\frac{\partial\ell(\boldsymbol{\theta};\boldsymbol{t},\boldsymbol{\delta})}{\partial\beta_0} = n_1 + \sum_{i=1}^{n}(1-\delta_i)\frac{\exp(D_i(\boldsymbol{\theta}))}{1+\exp(D_i(\boldsymbol{\theta}))} - \sum_{i=1}^{n}\frac{\exp(\mathbf{x}_i'\boldsymbol{\beta})}{1+\exp(\mathbf{x}_i'\boldsymbol{\beta})}.$$

$$\frac{\partial\ell(\boldsymbol{\theta};\boldsymbol{t},\boldsymbol{\delta})}{\partial\beta_1} = \sum_{i=1}^{n}\delta_i x_i + \sum_{i=1}^{n}(1-\delta_i)x_i\frac{\exp(D_i(\boldsymbol{\theta}))}{1+\exp(D_i(\boldsymbol{\theta}))} - \sum_{i=1}^{n}\frac{\exp(\mathbf{x}_i'\boldsymbol{\beta})x_i}{1+\exp(\mathbf{x}_i'\boldsymbol{\beta})}.$$

$$\begin{aligned}\frac{\partial\ell(\boldsymbol{\theta};\boldsymbol{t},\boldsymbol{\delta})}{\partial\gamma_1} = &-\frac{n_1}{\gamma_1} - \frac{n_1\log(\gamma_2)}{\gamma_1^2} - \frac{1}{\gamma_1^2}\sum_{i=1}^{n}\delta_i\log(t_i)\\ &+ \sum_{i=1}^{n}\delta_i\frac{\partial D_i(\boldsymbol{\theta})}{\partial\gamma_1} + \sum_{i=1}^{n}(1-\delta_i)\frac{\exp(D_i(\boldsymbol{\theta}))}{1+\exp(D_i(\boldsymbol{\theta}))}\frac{\partial D(\boldsymbol{\theta})}{\partial\gamma_1}.\end{aligned}$$

$$\frac{\partial \ell(\boldsymbol{\theta};\boldsymbol{t},\boldsymbol{\delta})}{\partial \gamma_2} = -\frac{n_1}{\gamma_1\gamma_2} + \sum_{i=1}^{n}\delta_i\frac{\partial D_i(\boldsymbol{\theta})}{\partial \gamma_2} + \sum_{i=1}^{n}(1-\delta_i)\frac{\exp(D_i(\boldsymbol{\theta}))}{1+\exp(D_i(\boldsymbol{\theta}))}\frac{\partial D_i(\boldsymbol{\theta})}{\partial \gamma_2}.$$

The 2nd derivatives:

$$\frac{\partial^2 \ell(\boldsymbol{\theta};\boldsymbol{t},\boldsymbol{\delta})}{\partial \beta_0^2} = \sum_{i=1}^{n}(1-\delta_i)\frac{\exp(D_i(\boldsymbol{\theta}))}{\{1+\exp(D_i(\boldsymbol{\theta}))\}^2} - \sum_{i=1}^{n}\frac{\exp(\mathbf{x}_i'\boldsymbol{\beta})}{\{1+\exp(\mathbf{x}_i'\boldsymbol{\beta})\}^2}.$$

$$\frac{\partial^2 \ell(\boldsymbol{\theta};\boldsymbol{t},\boldsymbol{\delta})}{\partial \beta_0\partial \beta_1} = \sum_{i=1}^{n}(1-\delta_i)\frac{\exp(D_i(\boldsymbol{\theta}))x_i}{\{1+\exp(D_i(\boldsymbol{\theta}))\}^2} - \sum_{i=1}^{n}\frac{\exp(\mathbf{x}_i'\boldsymbol{\beta})x_i}{\{1+\exp(\mathbf{x}_i'\boldsymbol{\beta})\}^2}.$$

$$\frac{\partial^2 \ell(\boldsymbol{\theta};\boldsymbol{t},\boldsymbol{\delta})}{\partial \beta_0\partial \gamma_1} = \sum_{i=1}^{n}(1-\delta_i)\frac{\exp(D_i(\boldsymbol{\theta}))}{\{1+\exp(D_i(\boldsymbol{\theta}))\}^2}\frac{\partial D_i(\boldsymbol{\theta})}{\partial \gamma_1}.$$

$$\frac{\partial^2 \ell(\boldsymbol{\theta};\boldsymbol{t},\boldsymbol{\delta})}{\partial \beta_0\partial \gamma_2} = \sum_{i=1}^{n}(1-\delta_i)\frac{\exp(D_i(\boldsymbol{\theta}))}{\{1+\exp(D_i(\boldsymbol{\theta}))\}^2}\frac{\partial D_i(\boldsymbol{\theta})}{\partial \gamma_2}.$$

$$\frac{\partial^2 \ell(\boldsymbol{\theta};\boldsymbol{t},\boldsymbol{\delta})}{\partial \beta_1^2} = \sum_{i=1}^{n}(1-\delta_i)\frac{\exp(D_i(\boldsymbol{\theta}))x_i^2}{\{1+\exp(D_i(\boldsymbol{\theta}))\}^2} - \sum_{i=1}^{n}\frac{\exp(\mathbf{x}_i'\boldsymbol{\beta})x_i^2}{\{1+\exp(\mathbf{x}_i'\boldsymbol{\beta})\}^2}.$$

$$\frac{\partial^2 \ell(\boldsymbol{\theta};\boldsymbol{t},\boldsymbol{\delta})}{\partial \beta_1\partial \gamma_1} = \sum_{i=1}^{n}(1-\delta_i)x_i\frac{\exp(D_i(\boldsymbol{\theta}))}{\{1+\exp(D_i(\boldsymbol{\theta}))\}^2}\frac{\partial D_i(\boldsymbol{\theta})}{\partial \gamma_1}.$$

$$\frac{\partial^2 \ell(\boldsymbol{\theta};\boldsymbol{t},\boldsymbol{\delta})}{\partial \beta_1\partial \gamma_2} = \sum_{i=1}^{n}(1-\delta_i)x_i\frac{\exp(D_i(\boldsymbol{\theta}))}{\{1+\exp(D_i(\boldsymbol{\theta}))\}^2}\frac{\partial D_i(\boldsymbol{\theta})}{\partial \gamma_2}.$$

$$\begin{aligned}\frac{\partial^2 \ell(\boldsymbol{\theta};\boldsymbol{t},\boldsymbol{\delta})}{\partial \gamma_1^2} &= \frac{n_1}{\gamma_1^2} + \frac{2n_1}{\gamma_1^3}\log(\gamma_2) + \frac{2}{\gamma_1^3}\sum_{i=1}^{n}\delta_i\log(t_i) + \sum_{i=1}^{n}\frac{\partial^2 D_i(\boldsymbol{\theta})}{\partial \gamma_1^2} \\ &\quad + \sum_{i=1}^{n}(1-\delta_i)\frac{\exp(D_i(\boldsymbol{\theta}))\left\{\left(\frac{\partial D_i(\boldsymbol{\theta})}{\partial \gamma_1}\right)^2 + \frac{\partial^2 D_i(\boldsymbol{\theta})}{\partial \gamma_1^2} + \frac{\partial^2 D_i(\boldsymbol{\theta})}{\partial \gamma_1^2}\exp(D_i(\boldsymbol{\theta}))\right\}}{\{1+\exp(D_i(\boldsymbol{\theta}))\}^2}.\end{aligned}$$

$$\frac{\partial^2 \ell(\boldsymbol{\theta}; \boldsymbol{t}, \boldsymbol{\delta})}{\partial \gamma_1 \partial \gamma_2} = \frac{-n_1}{\gamma_1^2 \gamma_2} + \sum_{i=1}^{n} \delta_i \frac{\partial^2 D_i(\boldsymbol{\theta})}{\partial \gamma_1 \partial \gamma_2} + \sum_{i=1}^{n} (1-\delta_i) \frac{\exp(D_i(\boldsymbol{\theta})) \left\{ \frac{\partial D_i(\boldsymbol{\theta})}{\partial \gamma_1} \frac{\partial D_i(\boldsymbol{\theta})}{\partial \gamma_2} + \frac{\partial^2 D_i(\boldsymbol{\theta})}{\partial \gamma_1 \partial \gamma_2} + \exp(D_i(\boldsymbol{\theta})) \frac{\partial^2 D_i(\boldsymbol{\theta})}{\partial \gamma_1 \partial \gamma_2} \right\}}{\{1 + \exp(D_i(\boldsymbol{\theta}))\}^2}.$$

$$\frac{\partial^2 \ell(\boldsymbol{\theta}; \boldsymbol{t}, \boldsymbol{\delta})}{\partial \gamma_2^2} = \frac{-n_1}{\gamma_1 \gamma_2^2} + \sum_{i=1}^{n} \delta_i \frac{\partial^2 D_i(\boldsymbol{\theta})}{\partial \gamma_2^2} + \sum_{i=1}^{n} (1-\delta_i) \frac{\exp(D_i(\boldsymbol{\theta})) \left\{ \left(\frac{\partial D_i(\boldsymbol{\theta})}{\partial \gamma_2} \right)^2 + \frac{\partial^2 D_i(\boldsymbol{\theta})}{\partial \gamma_2^2} + \frac{\partial^2 D_i(\boldsymbol{\theta})}{\partial \gamma_2^2} \exp(D_i(\boldsymbol{\theta})) \right\}}{\{1 + \exp(D_i(\boldsymbol{\theta}))\}^2}.$$

Appendix B: Derivatives of the log-likelihood under the G-gamma model

Recall from Equation (2) that the log-likelihood function is

$$\ell(\boldsymbol{\theta}) = \sum_{i=1}^{n} \delta_i \mathbf{x}_i' \boldsymbol{\beta} + \sum_{i=1}^{n} \delta_i \log f(t_i; \sigma, \lambda) + \sum_{i=1}^{n} (1-\delta_i) \log\{1 + \exp(\mathbf{x}_i' \boldsymbol{\beta}) S(t_i; \sigma, \lambda)\} - \sum_{i=1}^{n} \log\{1 + \exp(\mathbf{x}_i' \boldsymbol{\beta})\},$$

where $\boldsymbol{\theta} = (\boldsymbol{\beta}, \sigma, \lambda)$. We shall provide the derivative of $\ell(\boldsymbol{\theta})$ under $\boldsymbol{\beta} = (\beta_0, \beta_1)$.

The 1st derivatives:

$$\frac{\partial \ell}{\partial \beta_0} = \sum_{i=1}^{n} \delta_i + \sum_{i=1}^{n} (1-\delta_i) \frac{\exp(\mathbf{x}_i' \boldsymbol{\beta}) S(t_i; \gamma)}{\{1 + \exp(\mathbf{x}_i' \boldsymbol{\beta}) S(t_i; \gamma)\}} - \sum_{i=1}^{n} \frac{\exp(\mathbf{x}_i' \boldsymbol{\beta})}{1 + \exp(\mathbf{x}_i' \boldsymbol{\beta})},$$

$$\frac{\partial \ell}{\partial \beta_1} = \sum_{i=1}^{n} \delta_i x_i + \sum_{i=1}^{n} (1-\delta_i) x_i \frac{\exp(\mathbf{x}_i' \boldsymbol{\beta}) S(t_i; \gamma)}{\{1 + \exp(\mathbf{x}_i' \boldsymbol{\beta}) S(t_i; \gamma)\}} - \sum_{i=1}^{n} x_i \frac{\exp(\mathbf{x}_i' \boldsymbol{\beta})}{1 + \exp(\mathbf{x}_i' \boldsymbol{\beta})},$$

$$\frac{\partial \ell}{\partial \sigma} = \sum_{i=1}^{n} \delta_i \frac{\frac{\partial f(t_i;\gamma)}{\partial \sigma}}{f(t_i;\gamma)} + \sum_{i=1}^{n} (1-\delta_i) \frac{\exp(\mathbf{x}_i'\boldsymbol{\beta}) \frac{\partial S(t_i;\gamma)}{\partial \sigma}}{\{1+\exp(\mathbf{x}_i'\boldsymbol{\beta}) S(t_i;\gamma)\}},$$

$$\frac{\partial \ell}{\partial \lambda} = \sum_{i=1}^{n} \delta_i \frac{\frac{\partial f(t_i;\gamma)}{\partial \lambda}}{f(t_i;\gamma)} + \sum_{i=1}^{n} (1-\delta_i) \frac{\exp(\mathbf{x}_i'\boldsymbol{\beta}) \frac{\partial S(t_i;\gamma)}{\partial \lambda}}{\{1+\exp(\mathbf{x}_i'\boldsymbol{\beta}) S(t_i;\gamma)\}},$$

where with $q = 0.5$

$$\frac{\partial f(t_i;\gamma)}{\partial \sigma} = \frac{64(\lambda t_i)^{\frac{2}{\sigma}} \left(\log(\lambda t_i) \frac{-2}{\sigma} - 1) \right)}{3\sigma^2 t_i} \exp\{-4(\lambda t_i)^{\frac{1}{2\sigma}}\} + \frac{128(\lambda t_i)^{\frac{5}{2\sigma}}}{3\sigma^3 t_i} \exp\{-4(\lambda t_i)^{\frac{1}{2\sigma}}\} \log(\lambda t_i),$$

$$\frac{\partial S(t_i;\gamma)}{\partial \sigma} = \frac{-1}{2\sigma^2 \Gamma(4)} \log(\lambda t_i) \{4(\lambda t_i)^{\frac{0.5}{\sigma}}\}^4 \exp\{-4(\lambda t_i)^{\frac{0.5}{\sigma}}\},$$

$$\frac{\partial f(t_i;\gamma)}{\partial \lambda} = \exp\{-4(\lambda t_i)^{\frac{1}{2\sigma}}\} \frac{128(\lambda t_i)^{\frac{2}{\sigma-1}}}{3\sigma^2} \{1 - (\lambda t_i)^{\frac{1}{2\sigma}}\},$$

$$\frac{\partial S(t_i;\gamma)}{\partial \lambda} = \frac{1}{\Gamma(4)} \frac{128 t_i}{\sigma} (\lambda t_i)^{\frac{2}{\sigma-1}} \exp\{-4(\lambda t_i)^{\frac{0.5}{\sigma}}\}.$$

The 2nd derivatives:

$$\frac{\partial^2 \ell}{\partial \beta_0^2} = \sum_{i=1}^{n} (1-\delta_i) \frac{\exp(\mathbf{x}_i'\boldsymbol{\beta}) S(t_i;\gamma)}{\{1+\exp(\mathbf{x}_i'\boldsymbol{\beta}) S(t_i;\gamma)\}^2} - \sum_{i=1}^{n} \frac{\exp(\mathbf{x}_i'\boldsymbol{\beta})}{\{1+\exp(\mathbf{x}_i'\boldsymbol{\beta})\}^2},$$

$$\frac{\partial^2 \ell}{\partial \beta_0 \partial \beta_1} = \sum_{i=1}^{n} (1-\delta_i) x_i \frac{\exp(\mathbf{x}_i'\boldsymbol{\beta}) S(t_i;\gamma)}{\{1+\exp(\mathbf{x}_i'\boldsymbol{\beta}) S(t_i;\gamma)\}^2} - \sum_{i=1}^{n} x_i \frac{\exp(\mathbf{x}_i'\boldsymbol{\beta})}{\{1+\exp(\mathbf{x}_i'\boldsymbol{\beta})\}^2},$$

$$\frac{\partial^2 \ell}{\partial \beta_0 \partial \sigma} = \sum_{i=1}^{n} (1-\delta_i) \frac{\exp(\mathbf{x}_i'\boldsymbol{\beta})}{\{1+\exp(\mathbf{x}_i'\boldsymbol{\beta})S(t_i;\gamma)\}^2} \frac{\partial S(t_i;\gamma)}{\partial \sigma},$$

$$\frac{\partial^2 \ell}{\partial \beta_0 \partial \lambda} = \sum_{i=1}^{n} (1-\delta_i) \frac{\exp(\mathbf{x}_i'\boldsymbol{\beta})}{\{1+\exp(\mathbf{x}_i'\boldsymbol{\beta})S(t_i;\gamma)\}^2} \frac{\partial S(t_i;\gamma)}{\partial \lambda},$$

$$\frac{\partial^2 \ell}{\partial \beta_1^2} = \sum_{i=1}^{n} (1-\delta_i) x_i^2 \frac{\exp(\mathbf{x}_i'\boldsymbol{\beta})S(t_i;\gamma)}{\{1+\exp(\mathbf{x}_i'\boldsymbol{\beta})S(t_i;\gamma)\}^2} - \sum_{i=1}^{n} x_i^2 \frac{\exp(\mathbf{x}_i'\boldsymbol{\beta})}{\{1+\exp(\mathbf{x}_i'\boldsymbol{\beta})\}^2},$$

$$\frac{\partial^2 \ell}{\partial \beta_1 \partial \sigma} = \sum_{i=1}^{n} (1-\delta_i) x_i \frac{\exp(\mathbf{x}_i'\boldsymbol{\beta})}{\{1+\exp(\mathbf{x}_i'\boldsymbol{\beta})S(t_i;\gamma)\}^2} \frac{\partial S(t_i;\gamma)}{\partial \sigma},$$

$$\frac{\partial^2 \ell}{\partial \beta_1 \partial \lambda} = \sum_{i=1}^{n} (1-\delta_i) x_i \frac{\exp(\mathbf{x}_i'\boldsymbol{\beta})}{\{1+\exp(\mathbf{x}_i'\boldsymbol{\beta})S(t_i;\gamma)\}^2} \frac{\partial S(t_i;\gamma)}{\partial \lambda},$$

$$\begin{aligned}\frac{\partial^2 \ell}{\partial \sigma^2} =& \sum_{i=1}^{n} \delta_i \frac{\frac{\partial^2 f(t_i;\gamma)}{\partial \sigma^2} f(t_i;\gamma) - \frac{\partial f(t_i;\gamma)}{\partial \sigma}\frac{\partial f(t_i;\gamma)}{\partial \sigma}}{f(t_i;\gamma)^2} \\ &+ \sum_{i=1}^{n} (1-\delta_i) \frac{\exp(\mathbf{x}_i'\boldsymbol{\beta})\frac{\partial^2 S(t_i;\gamma)}{\partial \sigma^2}\{1+\exp(\mathbf{x}_i'\boldsymbol{\beta})S(t_i;\gamma)\} - \left\{\exp(\mathbf{x}_i'\boldsymbol{\beta})\frac{\partial S(t_i;\gamma)}{\partial \sigma}\right\}^2}{\{1+\exp(\mathbf{x}_i'\boldsymbol{\beta})S(t_i;\gamma)\}^2},\end{aligned}$$

$$\begin{aligned}\frac{\partial^2 \ell}{\partial \sigma \partial \lambda} =& \sum_{i=1}^{n} \delta_i \frac{\frac{\partial^2 f(t_i;\gamma)}{\partial \sigma \partial \lambda} f(t_i;\gamma) - \frac{\partial f(t_i;\gamma)}{\partial \sigma}\frac{\partial f(t_i;\gamma)}{\partial \lambda}}{f(t_i;\gamma)^2} \\ &+ \sum_{i=1}^{n} (1-\delta_i) \frac{\exp(\mathbf{x}_i'\boldsymbol{\beta})\frac{\partial^2 S(t_i;\gamma)}{\partial \sigma \partial \lambda}\{1+\exp(\mathbf{x}_i'\boldsymbol{\beta})S(t_i;\gamma)\} - \exp(\mathbf{x}_i'\boldsymbol{\beta})^2\frac{\partial f(t_i;\gamma)}{\partial \sigma}\frac{\partial S(t_i;\gamma)}{\partial \lambda}}{\{1+\exp(\mathbf{x}_i'\boldsymbol{\beta})S(t_i;\gamma)\}^2},\end{aligned}$$

$$\begin{aligned}\frac{\partial^2 \ell}{\partial \lambda^2} =& \sum_{i=1}^{n} \delta_i \frac{\frac{\partial^2 f(t_i;\gamma)}{\partial \lambda^2} f(t_i;\gamma) - \frac{\partial f(t_i;\gamma)}{\partial \lambda}\frac{\partial f(t_i;\gamma)}{\partial \lambda}}{f(t_i;\gamma)^2} \\ &+ \sum_{i=1}^{n} (1-\delta_i) \frac{\exp(\mathbf{x}_i'\boldsymbol{\beta})\frac{\partial^2 S(t_i;\gamma)}{\partial \lambda^2}\{1+\exp(\mathbf{x}_i'\boldsymbol{\beta})S(t_i;\gamma)\} - \left\{\exp(\mathbf{x}_i'\boldsymbol{\beta})\frac{\partial S(t_i;\gamma)}{\partial \lambda}\right\}^2}{\{1+\exp(\mathbf{x}_i'\boldsymbol{\beta})S(t_i;\gamma)\}^2},\end{aligned}$$

where with $q = 0.5$

$$\frac{\partial^2 f(t_i;\gamma)}{\partial\sigma^2} = \frac{128}{3t_i}\frac{\left(2+\log(\lambda t_i)\frac{2}{\sigma}\right)\log(\lambda t_i)+\left(\log(\lambda t_i)\frac{-2}{\sigma}-1\right)\left\{\log(\lambda t_i)(\lambda t_i)^{\frac{1}{2\sigma}}-\sigma\right\}}{\sigma^4}$$
$$\times(\lambda t_i)^{\frac{2}{\sigma}}\exp\{-4(\lambda t_i)^{\frac{1}{2\sigma}}\}$$
$$+\frac{128\log(\lambda t_i)}{3t_i}\frac{\left\{\frac{-5}{2}+2(\lambda t_i)^{\frac{1}{2\sigma}}\right\}\log(\lambda t_i)\frac{1}{\sigma}-3}{\sigma^4}\exp\{-4(\lambda t_i)^{\frac{1}{2\sigma}}\}(\lambda t_i)^{\frac{5}{2\sigma}},$$

$$\frac{\partial^2 S(t_i;\gamma)}{\partial\sigma^2} = \frac{-256\log(\lambda t_i)}{\Gamma(4)}\frac{\{(\lambda t_i)^{\frac{1}{2\sigma}}-1\}\log(\lambda t_i)-\sigma}{\sigma^4}\exp\{-4(\lambda t_i)^{\frac{0.5}{\sigma}}\}(\lambda t_i)^{\frac{2}{\sigma}},$$

$$\frac{\partial^2 f(t_i;\gamma)}{\partial\sigma\partial\lambda} = \left\{\left[(\lambda t_i)^{\frac{1}{2\sigma}}\log(\lambda t_i)-\log(\lambda t_i)-\sigma\right]2\{1-(\lambda t_i)^{\frac{1}{2\sigma}}\}+(\lambda t_i)^{\frac{1}{2\sigma}}\log(\lambda t_i)\frac{-1}{2}\right\}$$
$$\times\exp\{-4(\lambda t_i)^{\frac{1}{2\sigma}}\}\frac{128}{3\sigma^4}(\lambda t_i)^{\frac{2}{\sigma-1}},$$

$$\frac{\partial^2 f(t_i;\gamma)}{\partial\sigma\partial\lambda} = \left[\log(\lambda t_i)\frac{2}{\sigma}+2+\left(\log(\lambda t_i)\frac{-2}{\sigma}-1\right)(\lambda t_i)^{\frac{1}{2\sigma}}\right]\frac{-128}{3t_i\lambda\sigma^3}$$
$$\times\exp\{-4(\lambda t_i)^{\frac{1}{2\sigma}}\}(\lambda t_i)^{\frac{2}{\sigma}},$$

$$\frac{\partial^2 S(t_i;\gamma)}{\partial\sigma\partial\lambda} = \frac{128t_i}{\Gamma(4)}\frac{\log(\lambda t_i)\frac{-2}{\sigma}+(\lambda t_i)^{\frac{1}{2\sigma}}\log(\lambda t_i)\frac{2}{\sigma}-1}{\sigma^2}\exp\{-4(\lambda t_i)^{\frac{0.5}{\sigma}}\}(\lambda t_i)^{\frac{2}{\sigma-1}},$$

$$\frac{\partial f(t_i;\gamma)}{\partial\lambda} = \exp\{-4(\lambda t_i)^{\frac{1}{2\sigma}}\}\frac{128(\lambda t_i)^{\frac{2}{\sigma-1}}}{3\sigma^2}\{1-(\lambda t_i)^{\frac{1}{2\sigma}}\},$$

$$\frac{\partial^2 f(t_i;\gamma)}{\partial\lambda^2} = \frac{128}{3\sigma^2}\exp\{-4(\lambda t_i)^{\frac{1}{2\sigma}}\}(-2)\frac{1}{\lambda\sigma}(\lambda t_i)^{\frac{2.5}{\sigma-1}}\{1-(\lambda t_i)^{\frac{1}{2\sigma}}\}$$
$$+\frac{128}{3\sigma^2}\exp\{-4(\lambda t_i)^{\frac{1}{2\sigma}}\}\left\{(\frac{2}{\sigma}-1)\frac{1}{\lambda}\left\{1-(\lambda t_i)^{\frac{1}{2\sigma}}\right\}+\frac{-t_i}{2\sigma}(\lambda t_i)^{\frac{1}{2\sigma-1}}\right\}(\lambda t_i)^{\frac{2}{\sigma-1}},$$

$$\frac{\partial^2 S(t_i;\gamma)}{\partial\lambda^2} = \frac{1}{\Gamma(4)}\frac{128t_i}{\sigma}\exp\{-4(\lambda t_i)^{\frac{0.5}{\sigma}}\}(\lambda t_i)^{\frac{2}{\sigma-1}}\left\{\left(\frac{2}{\sigma}-1\right)\frac{1}{\lambda}+(-2)\frac{t_i}{\sigma}(\lambda t_i)^{\frac{1}{2\sigma-1}}\right\}.$$

Algorithm 2: The NR algorithm with the G-gamma lifetime

Instead of writing the primitive NR algorithm, one can use the R routine 'nlm' to maximize the log-likelihood. This routine performs a robust version of the NR algorithm (MacDonald 2014) with initial values, given by users. Since we could not find the universally good randomization schemes for the initial values, we arbitrary set the initial values $\beta_0^{(0)}=\beta_1^{(0)}=0$, $\widetilde{\gamma}_1^{(0)}=-0.4$, $\widetilde{\gamma}_2^{(0)}=-1.3$, where $\gamma_1^{(0)}=\exp(\widetilde{\gamma}_1^{(0)})$, $\gamma_2^{(0)}=\exp(\widetilde{\gamma}_2^{(0)})$ in our numerical studies. Under this setup, the nlm always ascertain the MLE without randomization to the initial values.

Appendix C: R codes for simulations (Weibull model, NR algorithm)

```
D = function(Beta,Gam){   (Beta[1]+Xi*Beta[2])-(Gam[2]*Ti)^(1/Gam[1]) }
#### log-likelihood
WB_l_func = function(Beta,Gam){
sum(d*(Beta[1]+Xi*Beta[2]))-n1*log(Gam[1])-sum(d*log(Ti))+(log(Gam[2])/Gam[1])*n1+(1/Gam[1])*sum
(d*log(Ti))+sum(d*(-(Gam[2]*Ti)^(1/Gam[1])))+sum((1-d)*log(1+exp(D(Beta,Gam))))-sum(log(1+exp
(Beta[1]+Xi*Beta[2])))
}
#### 1 st derivatives
WB_l_1st_Beta0_func = function(Beta,Gam){
n1+sum((1-d)*exp(D(Beta,Gam))/(1+exp(D(Beta,Gam))))-sum(exp(Beta[1]+Xi*Beta[2])/(1+exp(Beta[1]+
Xi*Beta[2])))
}
WB_l_1st_Beta1_func = function(Beta,Gam){
sum(d*Xi)+sum((1-d)*Xi*exp(D(Beta,Gam))/(1+exp(D(Beta,Gam))))-sum(Xi*exp(Beta[1]+Xi*Beta[2])/(1+
exp(Beta[1]+Xi*Beta[2])))
}

D_1_Gam1 = function(Beta,Gam){   (Gam[2]*Ti)^(1/Gam[1])*log(Gam[2]*Ti)/(Gam[1]^2) }
### 1 st derivatives
WB_l_1st_Gam1_func = function(Beta,Gam){
-n1/Gam[1]-n1*log(Gam[2])/(Gam[1]^2)-sum(d*log(Ti))/(Gam[1]^2)+sum(d*D_1_Gam1(Beta,Gam))+sum
((1-d)*exp(D(Beta,Gam))*D_1_Gam1(Beta,Gam)/(1+exp(D(Beta,Gam))))
}

D_1_Gam2 = function(Beta,Gam){   -(1/(Gam[1]*Gam[2]))*(Gam[2]*Ti)^(1/Gam[1]) }
### 1 st derivatives
WB_l_1st_Gam2_func = function(Beta,Gam){
n1/(Gam[1]*Gam[2])+sum(d*D_1_Gam2(Beta,Gam))+sum((1-d)*exp(D(Beta,Gam))*D_1_Gam2(Beta,Gam)
/(1+exp(D(Beta,Gam))))
}
### 2nd derivatives
WB_l_2nd_Beta0_func = function(Beta,Gam){
sum((1-d)*exp(D(Beta,Gam))/(1+exp(D(Beta,Gam)))^2)-sum(exp(Beta[1]+Xi*Beta[2])/(1+exp(Beta[1]+Xi
*Beta[2]))^2)
}
WB_l_2nd_Beta0Beta1_func = function(Beta,Gam){
sum(Xi*(1-d)*exp(D(Beta,Gam))/(1+exp(D(Beta,Gam)))^2)-sum(Xi*exp(Beta[1]+Xi*Beta[2])/(1+exp(Beta
[1]+Xi*Beta[2]))^2)
}
WB_l_2nd_Beta0Gam1_func = function(Beta,Gam){ sum((1-d)*exp(D(Beta,Gam))*D_1_Gam1(Beta,Gam)/(1+
exp(D(Beta,Gam)))^2) }
WB_l_2nd_Beta0Gam2_func = function(Beta,Gam){ sum((1-d)*exp(D(Beta,Gam))*D_1_Gam2(Beta,Gam)/(1+
exp(D(Beta,Gam)))^2) }
WB_l_2nd_Beta1_func = function(Beta,Gam){
sum(Xi^2*(1-d)*exp(D(Beta,Gam))/(1+exp(D(Beta,Gam)))^2)-sum(Xi^2*exp(Beta[1]+Xi*Beta[2])/(1+exp(
Beta[1]+Xi*Beta[2]))^2)
}
WB_l_2nd_Beta1Gam1_func = function(Beta,Gam){
  sum(Xi*(1-d)*exp(D(Beta,Gam))*D_1_Gam1(Beta,Gam)/(1+exp(D(Beta,Gam)))^2)
}
WB_l_2nd_Beta1Gam2_func = function(Beta,Gam){
  sum(Xi*(1-d)*exp(D(Beta,Gam))*D_1_Gam2(Beta,Gam)/(1+exp(D(Beta,Gam)))^2)
}
```

```
D_2_Gam1 = function(Beta,Gam){
  (Gam[2]*Ti)^(1/Gam[1])*log(Gam[2]*Ti)/(Gam[1]^3)*(-log(Gam[2]*Ti)/Gam[1]-2)
}
WB_l_2nd_Gam1_func = function(Beta,Gam){
n1/(Gam[1]^2)+2*n1/(Gam[1]^3)*log(Gam[2])+2/(Gam[1]^3)*sum(d*log(Ti))+sum(d*D_2_Gam1(Beta,Gam))+
sum((1-d)*exp(D(Beta,Gam))*(D_1_Gam1(Beta,Gam)^2+D_2_Gam1(Beta,Gam)+D_2_Gam1(Beta,Gam)*
exp(D(Beta,Gam)))/(1+exp(D(Beta,Gam)))^2)
}
D_2_Gam1Gam2 = function(Beta,Gam){   (Gam[2]*Ti)^(1/Gam[1])/(Gam[1]^2*Gam[2])*(1+log(Gam[2]*Ti)/
Gam[1]) }
WB_l_2nd_Gam1Gam2_func = function(Beta,Gam){
-n1/(Gam[1]^2*Gam[2])+sum(d*D_2_Gam1Gam2(Beta,Gam))+sum((1-d)*exp(D(Beta,Gam))*(D_1_Gam1
(Beta,Gam)*D_1_Gam2(Beta,Gam)+D_2_Gam1Gam2(Beta,Gam)+D_2_Gam1Gam2(Beta,Gam)*exp(D(Beta,Gam)))/
(1+exp(D(Beta,Gam)))^2)
}
D_2_Gam2 = function(Beta,Gam){   (1-1/Gam[1])*(Gam[2]*Ti)^(1/Gam[1])/(Gam[1]*Gam[2]^2) }
WB_l_2nd_Gam2_func = function(Beta,Gam){
-n1/(Gam[1]*Gam[2]^2)+sum(d*D_2_Gam2(Beta,Gam))+sum((1-d)*exp(D(Beta,Gam))*(D_1_Gam2(Beta,Gam)^2
+D_2_Gam2(Beta,Gam)+D_2_Gam2(Beta,Gam)*exp(D(Beta,Gam)))/(1+exp(D(Beta,Gam)))^2)
}
# score function
WB_Score_l_func = function(Beta,Gam){
  S_B0 = WB_l_1st_Beta0_func(Beta,Gam)
  S_B1 = WB_l_1st_Beta1_func(Beta,Gam)
  S_G1 = WB_l_1st_Gam1_func(Beta,Gam)
  S_G2 = WB_l_1st_Gam2_func(Beta,Gam)
  matrix(c(S_B0,S_B1,S_G1,S_G2),ncol = 1,byrow = T)
}
# hessian matrix
WB_Hess_l_func = function(Beta,Gam){
  H_B0 = WB_l_2nd_Beta0_func(Beta,Gam)
  H_B0B1 = WB_l_2nd_Beta0Beta1_func(Beta,Gam)
  H_B0G1 = WB_l_2nd_Beta0Gam1_func(Beta,Gam)
  H_B0G2 = WB_l_2nd_Beta0Gam2_func(Beta,Gam)
  H_B1 = WB_l_2nd_Beta1_func(Beta,Gam)
  H_B1G1 = WB_l_2nd_Beta1Gam1_func(Beta,Gam)
  H_B1G2 = WB_l_2nd_Beta1Gam2_func(Beta,Gam)
  H_G1 = WB_l_2nd_Gam1_func(Beta,Gam)
  H_G1G2 = WB_l_2nd_Gam1Gam2_func(Beta,Gam)
  H_G2 = WB_l_2nd_Gam2_func(Beta,Gam)
matrix(c(H_B0,H_B0B1,H_B0G1,H_B0G2,H_B0B1,H_B1,H_B1G1,H_B1G2,H_B0G1,H_B1G1,H_G1,H_G1G2,H_B0G2,H
_B1G2,H_G1G2,H_G2),ncol = 4,nrow = 4,byrow = T)
}

p0_func = function(x){   1/(1+exp(beta0_true+beta1_true*x))  }

#### Setting (i) for Table 3 ####
n=200; group = c(rep(1,55),rep(2,60),rep(3,45),rep(4,40))
#n=400; group = c(rep(1,110),rep(2,120),rep(3,90),rep(4,80))
#n=600; group = c(rep(1,165),rep(2,180),rep(3,135),rep(4,120))
beta0_true = -1.192
beta1_true = 0.573
gam1_true = 0.316
gam2_true = 0.179
alpha_rec = c(0.033,0.03,0.028,0.021)
```

```
P0 = c(p0_func(1),p0_func(2),p0_func(3),p0_func(4))
p = c(0.8,0.65,0.5,0.35)

#### Setting (ii) for Table 3 ####
n=200; group = c(rep(1,55),rep(2,60),rep(3,45),rep(4,40))
#n=400; group = c(rep(1,110),rep(2,120),rep(3,90),rep(4,80))
#n=600; group = c(rep(1,165),rep(2,180),rep(3,135),rep(4,120))
beta0_true = -0.038
beta1_true = 0.443
gam1_true = 0.316
gam2_true = 0.179
alpha_rec = c(0.021,0.021,0.018,0.010)
P0 = c(p0_func(1),p0_func(2),p0_func(3),p0_func(4))
p = c(0.5,0.4,0.3,0.2)

#### Setting (i+) for Table 5 ####
n=200; group = c(rep(1,55),rep(2,60),rep(3,45),rep(4,40))
#n=400; group = c(rep(1,110),rep(2,120),rep(3,90),rep(4,80))
#n=600; group = c(rep(1,165),rep(2,180),rep(3,135),rep(4,120))
beta0_true = -1.192
beta1_true = 0.573
gam1_true = 1.2
gam2_true = 1.2
alpha_rec = c(0.033,0.03,0.028,0.021)
P0 = c(p0_func(1),p0_func(2),p0_func(3),p0_func(4))
p = c(0.8,0.65,0.5,0.35)

#### Setting (ii+) for Table 5 ####
n=200; group = c(rep(1,55),rep(2,60),rep(3,45),rep(4,40))
#n=400; group = c(rep(1,110),rep(2,120),rep(3,90),rep(4,80))
#n=600; group = c(rep(1,165),rep(2,180),rep(3,135),rep(4,120))
beta0_true = -0.038
beta1_true = 0.443
gam1_true = 1.2
gam2_true = 1.2
alpha_rec = c(0.021,0.021,0.018,0.010)
P0 = c(p0_func(1),p0_func(2),p0_func(3),p0_func(4))
p = c(0.5,0.4,0.3,0.2)

###### Monte Carlo Simulation ######
set.seed(100)
N = 200 ### No. of repetitions
beta0 = beta1 = gam1 = gam2 = censsork = iter_NR = se_beta0 = se_beta1 = se_gam1 = se_gam2 =se_
gam1_tuta=se_gam2_tuta= c()
for (j in 1:N) {
  ### generate data
  Ti = d = numeric(n)
  for (i in 1:n)   {
    xi = group[i]
    P0i = 1/(1+exp(beta0_true+beta1_true*xi))
    ui = runif(1)
    d[i] = 1*(ui>P0i)
    C = rexp(1,alpha_rec[xi])
    W = rweibull(1,1/gam1_true,1/gam2_true)
    Ti[i] = (1-d[i])*C+d[i]*min(C,W)
```

```
    d[i] = d[i]*1*(W<C)
  }
  n1 = sum(d)
  Xi = group
  censsork[j] = 1-mean(d)
  ### Newton-Raphson algorithm
  beta0_0 = beta1_0 = 0
  gam1_0 = gam2_0= -2
  Betah = c(beta0_0,beta1_0)
  gammah = c(gam1_0,gam2_0)
  e = 0
  repeat  {
    e = e+1
    beta0_old = Betah[1]
    beta1_old = Betah[2]
    gam1_old = gammah[1]
    gam2_old = gammah[2]
    Theta = c(Betah,gammah)
    Hess = WB_Hess_l_func(Betah,exp(gammah))
    Score = WB_Score_l_func(Betah,exp(gammah))
    Theta = Theta - solve(Hess)%*%Score
    Betah = Theta[1:2]
    gammah = Theta[3:4]
    if(abs(Betah[1])>3){
      beta0_ran = beta0_0 + runif(1,-1,1)
      Betah = c(beta0_ran,Betah[2])
    }
    if(abs(Betah[2])>3){
      beta1_ran = beta1_0 + runif(1,-1,1)
      Betah = c(Betah[1],beta1_ran)
    }
    a = max(abs(Betah[1]-beta0_old), abs(Betah[2]-beta1_old),abs(gammah[1]-gam1_old),abs(gammah
    [2]-gam2_old))
    if(a < 0.001) break
  }
  V_WB = solve(-WB_Hess_l_func(Betah,exp(gammah)))
  beta0 = c(beta0,Betah[1] )
  beta1 = c(beta1,Betah[2] )
  gam1 = c(gam1,gammah[1] )
  gam2 = c(gam2,gammah[2] )
  iter_NR[j] = e
  se_beta0 = c(se_beta0,sqrt(V_WB[1,1]))
  se_beta1 = c(se_beta1,sqrt(V_WB[2,2]))
  se_gam1 = c(se_gam1,sqrt(V_WB[3,3]))
  se_gam2 = c(se_gam2,sqrt(V_WB[4,4]))
  se_gam1_tuta = c(se_gam1_tuta,sqrt((exp(gammah[1])^(-2))*V_WB[3,3]) )
  se_gam2_tuta = c(se_gam2_tuta,sqrt((exp(gammah[2])^(-2))*V_WB[4,4]) )

}

mean(iter_NR) ## AI, the average number of iterations

Est=c(beta0=mean(beta0),beta1=mean(beta1), gamma1=mean(exp(gam1)),gamma2=mean(exp(gam2)))
round(Est,3)
```

```
Bias=c(beta0=mean(beta0)-beta0_true,beta1=mean(beta1)-beta1_true,
       gamma1=mean(exp(gam1))-gam1_true,gamma2=mean(exp(gam2))-gam2_true)
round(Bias,3)

SD=c(beta0=sd(beta0),beta1=sd(beta1),gamma1=sd(exp(gam1)),gamma2=sd(exp(gam2)))
round(SD,3)

SE=c(beta0=mean(se_beta0),beta1=mean(se_beta1),gamma1=mean(se_gam1),gamma2=mean(se_gam2))
round(SE,3)

interval_beta0 = (beta0_true<beta0+1.645*se_beta0)*(beta0_true>beta0-1.645*se_beta0)
interval_beta1 = (beta1_true<beta1+1.645*se_beta1)*(beta1_true>beta1-1.645*se_beta1)
interval_gam1 = (gam1_true<exp(gam1+1.645*se_gam1_tuta))*(gam1_true>exp(gam1-1.645*se_gam1_tuta))
interval_gam2 = (gam2_true<exp(gam2+1.645*se_gam2_tuta))*(gam2_true>exp(gam2-1.645*se_gam2_tuta))
CI90=c(beta0=mean(interval_beta0),beta1=mean(interval_beta1),gamma1=mean(interval_gam1),gamma2=
mean(interval_gam2))
round(CI90,3)

interval_beta0 = (beta0_true<beta0+1.96*se_beta0)*(beta0_true>beta0-1.96*se_beta0)
interval_beta1 = (beta1_true<beta1+1.96*se_beta1)*(beta1_true>beta1-1.96*se_beta1)
interval_gam1 = (gam1_true<exp(gam1+1.96*se_gam1_tuta))*(gam1_true>exp(gam1-1.96*se_gam1_tuta))
interval_gam2 = (gam2_true<exp(gam2+1.96*se_gam2_tuta))*(gam2_true>exp(gam2-1.96*se_gam2_tuta))
CI95=c(beta0=mean(interval_beta0),beta1=mean(interval_beta1),
       gamma1=mean(interval_gam1),gamma2=mean(interval_gam2))
round(CI95,3)
```

References

[1] Balakrishnan N, Pal S (2015). An EM algorithm for the estimation of parameters of a flexible cure rate model with generalized gamma lifetime and model discrimination using likelihood- and information-based methods. *Computational Statistics*, 30: 151-189.

[2] Balakrishnan N, Pal S (2016). Expectation maximization-based likelihood inference for flexible cure rate models with Weibull lifetimes. *Stat Methods Med Res*, 25(4): 1535–1563.

[3] Boag JW (1949). Maximum likelihood estimates of the proportion of patients cured by cancer therapy. *Journal of the Royal Statistical Society.* Series B (Methodological), 11(1): 15-53.

[4] Conway RW, Maxwell WL (1962). A queuing model with state dependent services rates. *Journal of Industrial Engineering*, 12: 132-136.

[5] Cordeiro GM, Rodrigues J, de Castro M (2012). The exponential COM-Poisson distribution. *Stat Papers*, 53(3): 653-664.

[6] Cox DR, Oakes D (1984). Analysis of Survival Data, CRC Press, New York.

[7] De Castro M, Cancho VG, Rodrigues J (2010). A hands-on approach for fitting long-term survival models under the GAMLSS framework. *Comput Methods Programs Biomed*, 97(2): 168-77.

[8] Duchateau L Janssen P (2007). The Frailty Model, Springer, New York.

[9] Emura T, Matsui S, Chen HY (2019), compound. Cox: univariate feature selection and compound covariate for predicting survival, *Comput Methods Programs Biomed* 168: 21-37.

[10] Emura T, Chen YH (2018). Analysis of Survival Data with Dependent Censoring, Copula-Based Approaches, *JSS Research Series in Statistics*, Springer, Singapore.

[11] Emura T, Pan CH (2017). Parametric maximum likelihood inference and goodness-of-fit tests for dependently left-truncated data, a copula-based approach, Stat Pap, doi: 10.1007/s00362-017-0947-z.

[12] Emura T, Shiu SK (2016). Estimation and model selection for left-truncated and right-censored lifetime data with application to electric power transformers analysis. *Communications in Statistics-Simulation and Computation*, 45 (9): 3171-89.

[13] He Z (2017). Parametric likelihood inference with censored survival data under the COM-Poisson cure models. *National Central University Electronic Theses & Dissertations.*

[14] Hu YH, Emura T (2015). Maximum likelihood estimation for a special exponential family under random double-truncation. *Computational Statistics* 30: 1199-1229.

[15] Kim JM, Jung YS, Soderberg T (2009). Directional dependence of genes using survival truncated FGM type modification copulas. *Communications in Statistics-Simulation and Computation*, 38(7): 1470-1484.

[16] Knight K (2000). Mathematical Statistics. Chapman and Hall, Boca Raton.

[17] Lange K (1995). A gradient algorithm locally equivalent to the EM algorithm. *JRSSB*, 57: 425-437.

[18] MacDonald IL (2014). Does Newton–Raphson really fail? *Stat Methods Med Res*, 23(3): 308-311.

[19] Maller RA, Zhou X (1996). Survival Analysis with Long-Term Survivors, (1st ed.). Wiley, Chichester.

[20] McLachlan GJ and Krishman T (2008). The EM algorithm and extensions, (2nd ed.), John Wiley & Sons, Hoboken.

[21] Meng XL (2014). Response: Did Newton–Raphson really fail? *Stat Methods Med Res*, 23(3): 312-314.

[22] Nadarajah S (2009). Useful moment and CDF formulations for the COM–Poisson distribution. *Stat Papers*, 50: 617-22.

[23] Nelsen RB (2006). An Introduction to Copulas, 2nd Edition. Springer, New York.

[24] Rodrigues J, de Castro M, Cancho VG, Balakrishnan N (2009). COM–Poisson cure rate survival models and an application to a cutaneous melanoma data. *J of Stat Plan Infer*, 139: 3605–3611.

[25] Sellers KF (2012). A generalized statistical control chart for over- or underdispersed data. *Quality Reliability Engineering International*, 28(1): 59-65.

[26] Tsiatis A (1975). A nonidentifiability aspect of the problem of competing risks. *Proceedings of the National Academy of Sciences*, 72(1): 20-22.

[27] Van Wieringen WN, Kun D, Hampel R, Boulesteix AL (2009). Survival prediction using gene expression data: A review and comparison. *Comp Stat Data Anal*, 53: 1590-1603.

[Received April 2018; accepted July 2018.]

康威-馬克士威-泊松治癒率模型對於存活資料之計算觀點

何致晟[1]　　江村剛志[2,†]

[1]浙江省醫學科學院藥物研究所

[2]國立中央大學統計研究所

摘　要

康威-馬克士威-泊松分配 (Conway-Maxwell-Poisson) 可以用於描述存活資料中之治癒比例，基於此模型，在文獻中已有兩種最大概似估計量之計算方法被提出，一種方法爲 R gamlss 套件所使用的方法，其利用了對數概似函數之一階導數，另一種方法則是使用了完整資料概似函數之最大期望演算法。在本篇文章中，我們提出了一個穩健的牛頓-拉弗森演算法，其穩健性是來自於對起始值之隨機擾動，以及對恆正參數之對數轉換，在伯努力治癒模型下，我們提供了概似函數之導數表示法與電腦程式碼給讀者使用。由於牛頓-拉弗森演算法使用了概似函數之一、二階導數，故其收斂速度較 R gamlss 套件快，同時我們也回顧了最大期望演算法，並利用模擬分析將其表現與牛頓-拉弗森演算法做比較，除此之外，我們還提出了一筆新的資料來配適康威-馬克士威-泊松治癒模型，並且討論使用兩種演算法之結果。

關鍵詞: 最大期望演算法、廣義伽瑪分配、牛頓-拉弗森演算法、存活分析、韋伯分配。

JEL classification: C13, C46.

[†]通訊作者: 江村剛志
E-mail: takeshiemura@gmail.com