Appendix to Variable selection in a flexible parametric mixture cure model with interval-censored data

Sylvie Scolas*[1], Anouar El Ghouch [1], Catherine Legrand [1], and Abderrahim Oulhaj [2]

[1]Institute of Statistics, Biostatistics and Actuarial Sciences (ISBA), Université catholique de Louvain, Louvain-la-Neuve, Belgium
[2]Institute of public health, College of Medicine & Health Sciences, United Arab Emirates University (UAEU), United Arab Emirates (UAE)

In this appendix, we show the results of further simulations, and also provide the R code we used. The first purpose of the simulations is to discuss the use of a model taking a fraction of cured individuals into account. The second one is to analyze the evolution of the bias and MSE of estimates, using the EGG-AFT mixture cure model, with respect to the evolution of the cure rate and the right-censoring proportion. Indeed, as stated in the main paper, we expect a decreasing MSE in latency if the cure rate increases and the right-censoring proportion stays the same. We also show that the use of naive initial values in the optimization process may lead to poor results, and report the number of times the variable selection algorithm failed to converge.
In Section 1, we briefly describe the data generation. Section 2 presents results of simulations using an EGG-AFT model without considering any cured individuals in the data. The comparison over several couples of cured rate and right-censoring proportion is covered in Section 3. Section 4 gives results of simulations concerning initial values, and Section 5 reports the number of failures from simulations concerning variable selection in the main paper. We end with the R-code in Section 6.

1 Data Generation

To be able to compare different couples of proportion of cure and right-censoring, we generate several scenarios (Table 1). Similarly to the main paper, we generate survival times for susceptibles such that:

$$\begin{aligned} \log(T) &= \beta X + \exp(\alpha)\varepsilon \\ &= 0.6X + \exp(0)\varepsilon \end{aligned} \tag{1}$$

*sylvie.scolas@uclouvain.be

with $X \sim \text{Bern}(0.5)$ and ε follows the EGG distribution with parameter $q = 0.5$. The cure variable $Y \sim \text{Bern}(p)$ and

$$p(Z) = \frac{\exp(\gamma_0 + \gamma_1 Z)}{1 + \exp(\gamma_0 + \gamma_1 Z)} \tag{2}$$

with Z = X. The right-censoring distribution is exponential with parameter $1/\nu$. Values of γ_0, γ_1 and ν are given in Table 1. Each uncensored time is then interval-censored: the first visit time is simulated from a uniform distribution on $[0, 0.5]$. As long as the event has not occurred, we generate new visits from a uniform on $[0.25, 0.5]$. Results are based on 2000 replications with a sample size of 500.

2 Simulations results: model without taking cure into account

When the cure proportion in the population is very low, one could expect that a model that does not take cure into account will perform almost as good as a mixture cure model. Of course, one will also expect that this model will deteriorate, in terms of bias and MSE, if the cure rate becomes larger. Table 2 contains results of simulations using the EGG-AFT model without cure. The right-censoring proportion is fixed at 60%, whereas the cure proportion increases from 10% to 60%. Clearly, we see that bias and MSE increases when the cure proportion increases.

We can compare these results with those of Table 3, which gives bias and MSE of the estimates when using the mixture cure model. The upper part of the Table gives results for 60% right-censoring, along with a cure rate varying from 10% to 60% as well. We see that the mixture cure model always performs better for the AFT parameters. Even when the cure proportion is low, such as 10%, the bias and MSE can be up to four times larger when using the model that does not take cure into account. In Figure 1, we give boxplots of corresponding estimates for 10% cure and 20% cure: above is the EGG-AFT mixture cure model and below the model without taking cure into account.

3 Simulations results: analysis of the cure and right-censoring proportions

In the main paper, results of simulations show that when the cure fraction is low compared to the right-censoring proportion, the bias and MSE in incidence can be large. Also, they show that for a given right-censoring proportion, an increasing cure rate leads to better results.

Table 3 shows bias and MSE for 60% to 10% right-censoring, along with a cure proportion varying from 10% to 20%, 30%, 40%, 50% or 60%. The table allows to compare bias for a same right-censoring proportion, and for a same cure proportion. First, for a same right-censoring proportion, for example, 60%: from left to right, i.e. from a lower to a larger cure rate, we see that the bias and MSE are decreasing for all parameters. The same trend is detected for every other proportion of right-censored individuals.

Second, if we focus on a cure rate, for example 10%, we see that the bias and MSE are decreasing if the right-censoring proportion decreases as well. This is also the same conclusion

for every other cure rate.
In conclusion, when using a mixture cure model, the more right-censored individuals are effectively cured, the more accurate the results will be, in terms of bias and MSE. Of course, this conclusion is not correct if the analysis does not take into account the fraction of cured. As seen in Section 2, this would rather be the opposite.
These results emphasize the trade-off to make between the bias in β's when not using a mixture cure model when appropriate; and the instability of the γ's estimates if the cure proportion is too low, when using a mixture cure model.

4 Studying the impact of initial values

As with any optimization problem, a good choice of initial values may be important. If they are not close enough to the solution, the algorithm may fail to converge. With the use of such a complex model as the EGG-AFT mixture cure model, naive initial value, such as 0 everywhere except for parameters q and α, initialized to 1, can give poor results. In our simulations, we use the following simple, but effective, initial values: β_{init} is solution to the classic linear regression with response $\log(T)$, and covariates $\boldsymbol{X}$; γ_{init} is solution to the classic logistic regression with response being the status (right-censored or not) and covariates $\boldsymbol{Z}$; α_{init} can be set to 1. The initial value for q_{init}, the parameter of the EGG distribution, can be chosen as 1 or -1, depending on the skewness of the response $\log(T)$. To illustrate the difference, we compare results using the dataset from the third scenario from the main paper. Table 4 gives results for three different forms of initial values: first, the true value, then the proposed ones, and finally the naive ones. We see that the proposal is very similar to results using true value as initial values, and that the naive solution results in greater MSE, specially in the incidence part.

5 Non-convergence

In the main paper, we report results of simulations concerning variable selection using the adaptive LASSO. For some scenarios, the algorithm failed to converge. In Table 5, we report, for each scenario and each sample size, the number of failures experienced.

Table 1: For each desired cure proportion, the table gives corresponding values of (γ_0, γ_1); along with the values of ν to reach each right-censoring proportion.

(γ_0, γ_1)	Cure Rate	10% RC	20% RC	30% RC	40% RC	50% RC	60% RC
(1.7,1)	10% Cure	100	13	6.5	4	2.2	1.5
(1,0.8)	20% Cure		100	10	6	3	2
(0.4,1)	30% Cure			40	9	4.5	2.5
(-0.5,2)	40% Cure				50	8	3.5
(-0.75,1.5)	50% Cure					60	6.5
(-1.5,2)	60% Cure						25

Table 2: Bias and MSE of estimates using a EGG-AFT model without cure, for 2000 replications.

	60% RC											
	10% Cure		20% Cure		30% Cure		40% Cure		50% Cure		60% Cure	
q	0.665	0.473	1.134	1.315	1.646	2.738	2.418	5.893	3.013	9.139	4.401	19.54
β	0.140	0.035	0.201	0.058	0.325	0.126	0.679	0.483	0.681	0.494	1.004	1.056
α	-0.207	0.048	-0.347	0.125	-0.455	0.212	-0.514	0.271	-0.632	0.407	-0.752	0.577

Table 3: Bias and MSE with the EGG-AFT Mixture Cure Model, for several right-censoring and cure rates.

	10% Cure		20% Cure		30% Cure		40% Cure		50% Cure		60% Cure	
	60 % RC											
q	-0.167	0.120	-0.187	0.135	-0.171	0.123	-0.135	0.099	-0.085	0.054	-0.068	0.041
β	-0.077	0.030	-0.048	0.028	-0.045	0.028	-0.044	0.023	-0.037	0.019	-0.030	0.016
α	0.010	0.018	0.026	0.020	0.022	0.018	0.005	0.013	-0.012	0.007	-0.017	0.004
γ_0	0.724	3.291	0.209	0.362	0.087	0.093	0.036	0.031	0.010	0.024	-0.010	0.029
γ_1	1.573	12.95	0.926	5.708	0.310	1.549	0.187	0.849	-0.001	0.061	0.009	0.051
	50 % RC											
q	-0.123	0.073	-0.115	0.066	-0.092	0.047	-0.072	0.039	-0.048	0.026		
β	-0.051	0.021	-0.038	0.020	-0.035	0.018	-0.031	0.014	-0.021	0.012		
α	0.005	0.011	0.006	0.010	-0.001	0.007	-0.010	0.005	-0.016	0.003		
γ_0	0.268	0.690	0.082	0.081	0.025	0.029	0.009	0.020	-0.002	0.019		
γ_1	1.468	10.32	0.284	1.683	0.027	0.121	0.011	0.076	0.004	0.039		
	40 % RC											
q	-0.075	0.034	-0.068	0.028	-0.060	0.026	-0.044	0.022				
β	-0.030	0.013	-0.028	0.013	-0.027	0.012	-0.019	0.010				
α	-0.002	0.005	-0.005	0.004	-0.009	0.003	-0.014	0.002				
γ_0	0.080	0.091	0.026	0.033	0.007	0.021	0.001	0.017				
γ_1	0.662	4.409	0.023	0.152	0.003	0.065	0.005	0.047				
	30 % RC											
q	-0.058	0.022	-0.051	0.020	-0.042	0.017						
β	-0.024	0.010	-0.023	0.010	-0.018	0.009						
α	-0.006	0.003	-0.009	0.003	-0.011	0.002						
γ_0	0.045	0.054	0.013	0.026	0.001	0.017						
γ_1	0.209	1.327	0.008	0.087	0.002	0.046						
	20 % RC											
q	-0.042	0.015	-0.034	0.013								
β	-0.019	0.008	-0.015	0.008								
α	-0.009	0.002	-0.011	0.002								
γ_0	0.024	0.041	0.006	0.021								
γ_1	0.042	0.234	0.005	0.057								
	10 % RC											
q	-0.032	0.011										
β	-0.015	0.007										
α	-0.010	0.001										
γ_0	0.017	0.032										
γ_1	0.013	0.115										

Table 4: Results of 2000 simulations: bias and MSE for the three different types of initial values.

Parameter	True Value	True Value Init		Our Proposal		Naive Proposal	
q	1	-0,043	0,085	-0,043	0,085	0,052	0,204
β_0	4.1	-0,002	0,001	-0,002	0,001	-0,004	0,001
β_1	-0.2	-0,001	0,001	-0,001	0,001	-0,009	0,004
β_2	0.5	-0,006	0,009	-0,006	0,009	-0,005	0,009
β_3	-0.5	0,002	0,004	0,002	0,004	0,002	0,004
γ_0	0.85	-0,035	0,159	-0,035	0,158	-4,208	604,5
γ_1	-0.85	0,032	0,175	0,032	0,174	0,033	94,88
γ_2	0.2	-0,005	0,004	-0,005	0,004	-0,291	23,66
γ_3	-0.4	0,000	0,264	0,000	0,264	-0,078	536,1
α_0	-2	0,040	0,026	0,040	0,026	-0,012	0,057
α_1	0.5	-0,008	0,017	-0,008	0,017	-0,013	0,023

Table 5: Number of failure (percentage of failures) over 2000 replications.

Sample Size	(20% Cure, 40% RC)		(30% Cure, 40% RC)		(40% Cure, 60% RC)	
$n = 200$	54	(2,7%)	1	(0,1%)	19	(1,0%)
$n = 300$	24	(1,2%)	0	(0,0%)	3	(0,2%)
$n = 500$	0	(0,0%)	0	(0,0%)	0	(0,0%)

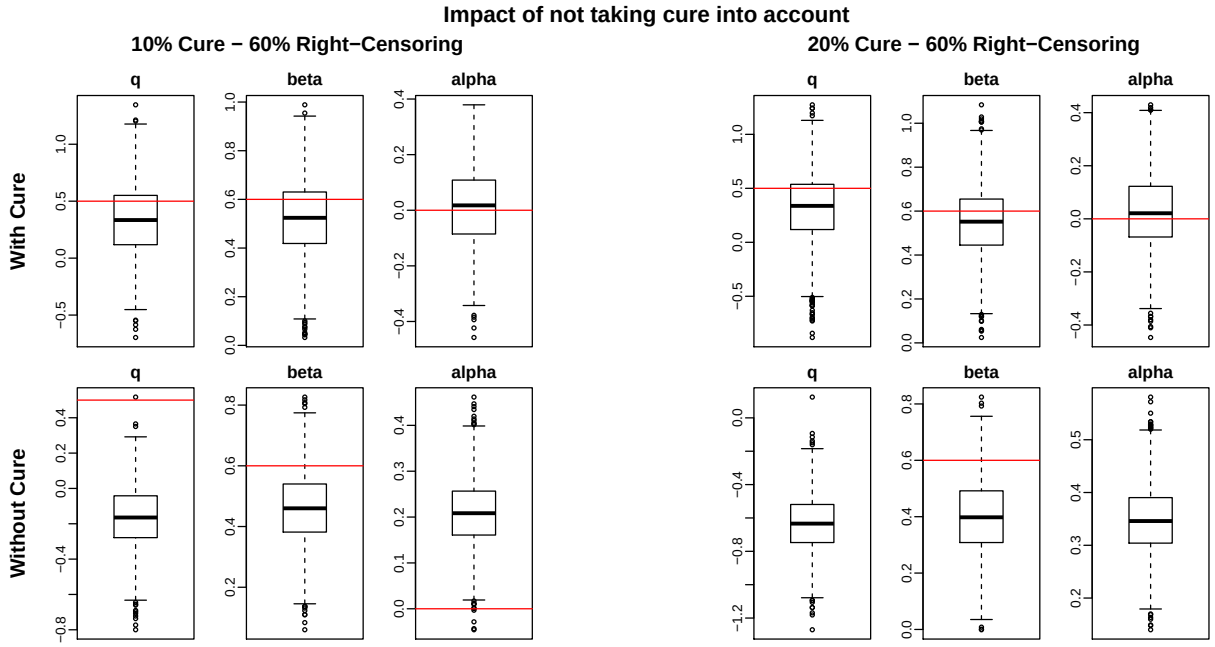

Figure 1: Boxplots of estimates for 10% cure and 20%, along with 60% RC. Above, using the EGG-AFT mixture cure model; below the EGG-AFT model without cure. The horizontal line represents the true value of the parameter.

6 R code

In this section, we include the R-code we use to get estimates for our EGG-AFT mixture cure model, with adaptive LASSO.

```
regg =function(n, q, sigma, mu){
  #Generate EGG-distributed random variables
  if (q!=0){
    qs <- q/sigma
    lbd <- exp(-qs*mu)/q^2
    (rgamma(n,q^(-2))/lbd)^(1/qs)
    # E0=rgamma(n,q^(-2)); E=log(q^2*E0)/q; X=exp(mu+sigma*E)
  }
  else rlnorm(n,mu,sigma) # Y=rnorm(n); X=exp(mu+sigma*Y)
}
Segg=function (t, q, sigma, mu) {
  # Survival distribution of the EGG.
  if (q != 0) {
    qs <- q/sigma
    lbd <- exp(-qs * mu)/q^2
    if (q > 0) 1-pgamma(lbd * t^qs, q^(-2))
    else pgamma(lbd * t^qs, q^(-2))  }
  else 1-plnorm(t, mu, sigma) }
```

```
fegg=function (t, q, sigma, mu){
  #Density function of the EGG
  if (q != 0) {
    qs <- q/sigma
    lbd <- exp(-qs * mu)/q^2
    newt <- t^qs
    dgamma(lbd * newt, q^(-2)) * lbd * abs(qs) * t^(qs-1)
  }
  else dlnorm(t, mu, sigma)
}
loglik = function(param, X, Z1, Z2, L, R, event,fegg,Segg){
  # Likelihood function for interval censoring , right
  # censoring, left censoring and cure.
  # param = c(q, beta, gnamma, alpha) as in the paper
  q = as.vector(param[1])
  X = as.matrix(X); p = ncol(X)
  beta= as.vector(param[2:(p+1)])
  muX = as.vector(X%*%beta)
  n = nrow(as.matrix(L))
  if (is.null(Z1)) {s1=0; pcureZ1 = rep(1,n)} else
  {Z1 = as.matrix(Z1); s1= ncol(Z1) #if Z1=1 pcure=constant;
   gama= as.vector(param[(p+2):(p+s1+1)])
   gamaZ1= as.vector(Z1%*%gama)
   pcureZ1 = exp(gamaZ1)/(1+exp(gamaZ1))}
  if(is.null(Z2)) {s2=0; sigmaZ2=param[p+s1+2]} else
{Z2 = as.matrix(Z2); s2= ncol(Z2) #if Z2=1 var=constant;
 alpha= as.vector(param[(p+s1+2):(p+s1+s2+1)])
 sigmaZ2 =as.vector(exp(Z2%*%alpha))}
fl=fegg(L,q,sigmaZ2,muX)
Sr=Segg(R,q,sigmaZ2,muX)
Sl=Segg(L,q,sigmaZ2,muX)
L.E=(pcureZ1*fl)[event==1]
L.LC=(pcureZ1*(1-Sr))[event==2]
L.IC=(pcureZ1*(Sl-Sr))[event==3]
L.RC=((pcureZ1*Sl)+(1 - pcureZ1))[event==0]
logLik =sum(log(L.E))+sum(log(L.LC))+sum(log(L.IC))+sum(log(L.RC))
return(-logLik)
}
loglik_fixQ = function(param,q, X, Z1, Z2, L, R, event, fegg, Segg){
  # EGG-AFT mixture cure likelihood, with fixed q,
  # to test for weibull (q=1) or lognormal(q=0)
  n = nrow(as.matrix(L))
  paramQ = c(q, param)
  ll = loglik(paramQ,X, Z1, Z2, L, R, event, fegg,Segg)
  ll
```

```
}
loglik_fixQ_Alpha = function(param,q,alpha, X, Z1, Z2, L, R, event, fegg, Segg){
  # EGG-AFT mixture cure likelihood, with fixed q AND alpha,
  # to test for exponential (q=1, alpha=0)
  n = nrow(as.matrix(L))
  paramQ = c(q, param,alpha)
  ll = loglik(paramQ,X, Z1, Z2, L, R, event, fegg,Segg)
  ll
}
initVal = function(X, Z,Xsigma, L, R, event,lasso=FALSE){
  #Gives initial values
  n=length(event)
  response =ifelse(L==0, R, L)
  status = event
  if(any(event==3) || (event==2)){
    response = ifelse(event==3, (L+R)/2, response)
    status = ifelse(event==0, 0 ,1)
  }

  inits.beta = if(!is.null(X)) lm(log(response)~X[,-1])$coef else c()
  inits.gamma = if(!is.null(Z)) glm(status~Z[,-1], family ="binomial")$coef else c()

  inits.param = c(1, inits.beta,inits.gamma, rep(1, ncol(as.matrix(Xsigma))))
  inits.param
}


#-------------------------------- ADAPTIVE LASSO-----------------------------#
require(lars)
lsaS <- function(intercept, n, Sigma, beta.ols)
{  #LSA version allowing "our" log-likelihood.
  # You NEED to run to original LARS variant for LSA from [Wang&Leng,2007].
  t1 <- sort(l.fit$BIC, index.return=TRUE)
  t2 <- sort(l.fit$AIC, index.return=TRUE)
  beta <- l.fit$beta
  if(intercept) {
    beta0 <- l.fit$beta0+beta.ols[1]
    beta.bic <- c(beta0[t1$ix[1]],beta[t1$ix[1],])
    beta.aic <- c(beta0[t2$ix[1]],beta[t2$ix[1],])
  }
  else {
    beta0 <- l.fit$beta0
    beta.bic <- beta[t1$ix[1],]
    beta.aic <- beta[t2$ix[1],]
    beta.bic2 = beta[t1$ix[2],]
```

```
  }
  obj <- list(beta.ols=beta.ols, beta.bic=beta.bic,
              beta.aic = beta.aic, BIC= l.fit$BIC,
              dof = l.fit$dof,all.beta =beta )
  obj
}

# --------- Adaptive Lasso for our mixture cure model -------------------------#
DoubleLassoDoubleLik = function(inits, X,Z1,Z2,L,R,event,fegg,Segg,method){
  ##############################################################################
  #   Input :
  #     inits : the initial value for the likelihood optimisation
  #     X  : the covariates to be included in the latency part
  #     Z1 : the covariates to be included in the incidence part
  #     Z2 : the covariates to be inclided in the scale part
  #     L  : the left bound of the interval for interval censored events
  #     R  : the right bound of the interval for interval censored events
  #     event : indicator variable (0 if RC, 1 if event, 3 if IC)
  #     fegg : egg density distribution
  #     Segg : egg survival distribution
  #
  #   Output :
  #     coefficients estimations with exact zero
  #
  #   Method :
  #   This function pereforms variable selection and estimation through
  #   the alasso for a mixture cure model and a likelihood 'loglik'
  #   (here: egg likelihood).
  #   It procedes iteratively with 2 steps : first a adaptive lasso
  #   on latency part (i.e. on the columns of the hessian of the likelihood
  #   for the 'X' matrix), then on the incidence part (i.e. on the columns
  #   of the hessian of the likelihood for the Z1 matrix). No shrinkage is
  #   done for intercepts and scale part.
  #
  ##############################################################################

  X = as.matrix(X); Z1 = as.matrix(Z1); Z2 = as.matrix(Z2)
  Xor =X; X = scale(X, F, T)               #Scaling X
  scaleX = attr(X,"scaled:scale")
  Z1or = Z1;Z1 = scale(Z1, FALSE, TRUE)    #Scaling Z1
  scaleZ1 = attr(Z1,"scaled:scale")
  p=ncol(X)                                # Number of covariates in latency
  s=ncol(Z1)                               # Number of covariates in incidence
  v = ncol(Z2)                             # Number of covariates in location
  n=nrow(X)                                # Number of observations
```

```
#-------Iteration
eps = 0.0001
delta = 1
conv = 0
inits = inits*c(1, scaleX, scaleZ1, rep(1, ncol(Z2)))

optim.obj =optim(inits,loglik,X=X,Z1=Z1,Z2=Z2,L=L,R=R,event=event,fegg=fegg,
       Segg=Segg,hessian = TRUE,method = method, control = list(maxit = 2000))
p.optim = optim.obj$par
q.optim = p.optim[1]
beta.optim= p.optim[(2):(p+1)]
gamma.optim = p.optim[(p+2):(p+s+1)]
alpha.optim = p.optim[(p+s+2) : length(p.optim)]
beta.lsa.old = beta.optim[-1]
gamma.lsa.old = gamma.optim[-1]
while(delta>eps ){
  #Beta-Part
  optim.Lat = optim(beta.optim,likBeta,param=c(q.optim, gamma.optim,alpha.optim),
                    X=X,Z=Z1,ZZ=Z2,L=L,R=R,event=event,fegg=fegg,Segg=Segg,
                    hessian=T,method = method,control = list(maxit = 1000))
  beta.par = optim.Lat$par;ddf.Lat = optim.Lat$hessian
  lsa.obj.Lat = lsaS(intercept=F,n=n,
        Sigma=solve(ddf.Lat[-1,-1]),beta.ols=beta.par[-1])
  beta.lsa = lsa.obj.Lat$beta.bic
  #Gamma-Part
  optim.Inc = optim(gamma.optim,likGamma,param=c(q.optim,beta.optim,alpha.optim),
                    X=X,Z=Z1,ZZ=Z2,L=L,R=R,event=event,fegg=fegg,Segg=Segg,
                    hessian=T,method = method,control = list(maxit = 1000))
  gamma.par = optim.Inc$par;ddf.Inc = optim.Inc$hessian
  lsa.obj.Inc = lsaS(intercept=F,n=n,
        Sigma=solve(ddf.Inc[-1,-1]),beta.ols=gamma.par[-1])
  gamma.lsa = lsa.obj.Inc$beta.bic

  inits =c(q.optim, beta.par[1],beta.lsa, gamma.par[1],gamma.lsa, alpha.optim)
  optim.obj =optim(inits,loglik,X=X,Z1=Z1,Z2=Z2,L=L,R=R,event=event,fegg=fegg,
              Segg=Segg, hessian = TRUE,method =method,control = list(maxit = 1000))
  p.optim = optim.obj$par
  q.optim.new = p.optim[1]
  alpha.optim.new = p.optim[(p+s+2) : length(p.optim)]
  delta = max(c(abs(q.optim-q.optim.new),abs(beta.lsa-beta.lsa.old),
                abs(gamma.lsa-gamma.lsa.old), alpha.optim.new-alpha.optim))
  beta.optim =beta.par
  beta.lsa.old= beta.lsa
  gamma.optim= gamma.par
  gamma.lsa.old = gamma.lsa
```

```
    q.optim= q.optim.new
    alpha.optim = alpha.optim.new
    inits =c(q.optim, beta.par[1],beta.lsa, gamma.par[1],gamma.lsa, alpha.optim)
    conv = conv+1
    if(conv ==25) stop("non convergence")
  }
  return(inits/c(1, scaleX, scaleZ1, rep(1, ncol(Z2))))
}
```