A connection between accelerated failure time model and $\lambda = aw^{b}$ model (Draft)

By Xintong Li (Gary)

Question

Statisticians like to use the logarithm to transform the data when coping with survival or failure time data. In the SAS, its default regression algorithm for the parametric distribution is also utilized a logarithm transformation. So, why we like the logarithm transformation and what’s the internal algorithm for the SAS when coping with the survival data. This is what the author tries to explore in this report.

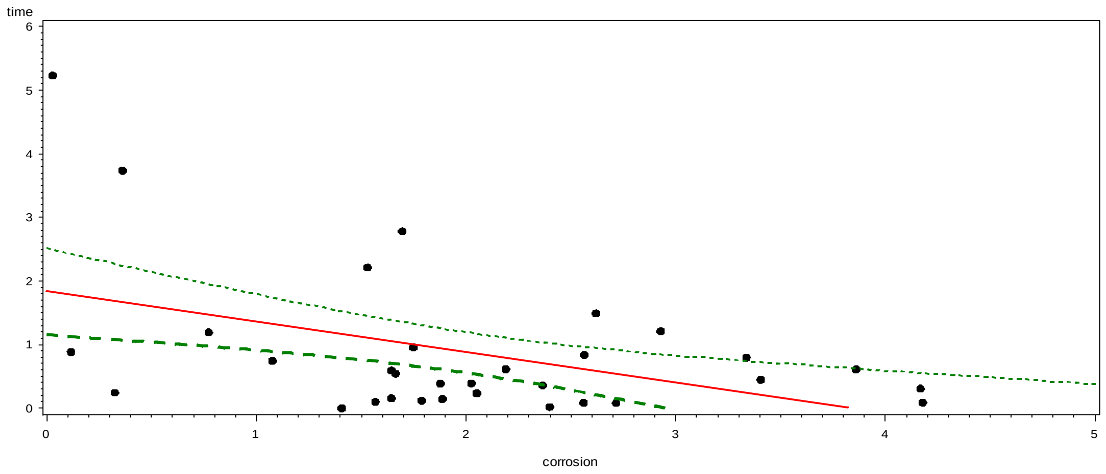
Introduce of the data and graphics

The dataset used in this article is named "engine", which is from the R library "ismev". It contains two variables and 32 observations, life span of a component, which designated to "Time" and corrosion degree of the component, which designated to "Corrosion". Here are some values in the dataset:

	Time	Corrosion
1	5.231237563	0.02856561
2	0.883741162	0.11644553
3	0.245824519	0.32556412
……..		
32	0.089619767	4.17895697

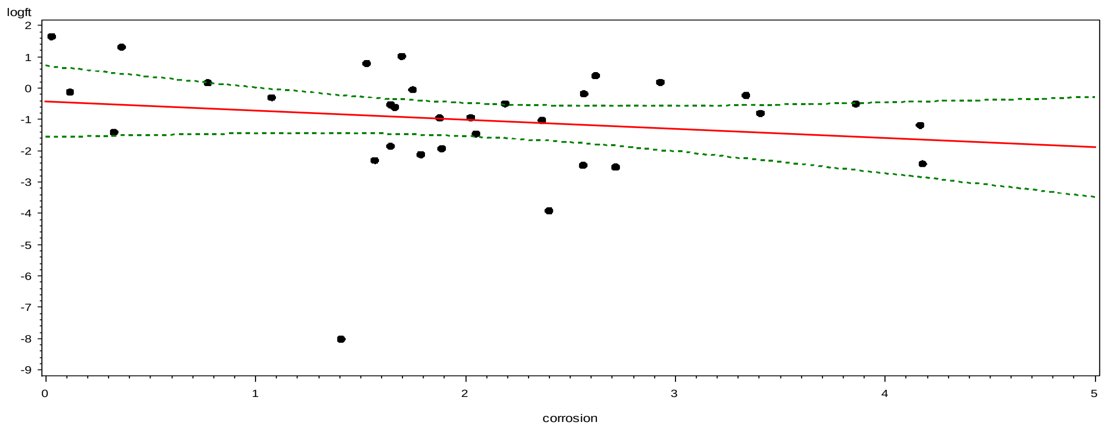
It is reasonable to assume that degree of corrosion will effect life span of them. So, we plan to use a regression to construct a model between them.

At first, we plot a graph of time vs corrosion with linear regression.

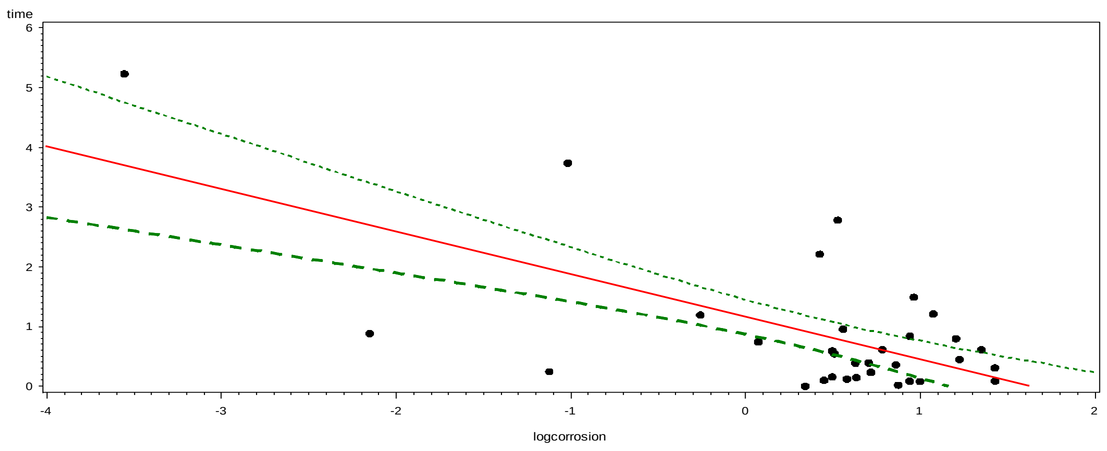


Clearly, the linear regression doesn't fit well, because many of the observed values fall out of regression prediction region. Then, three other transformed models were plotted to try to fix the issue of model fit.

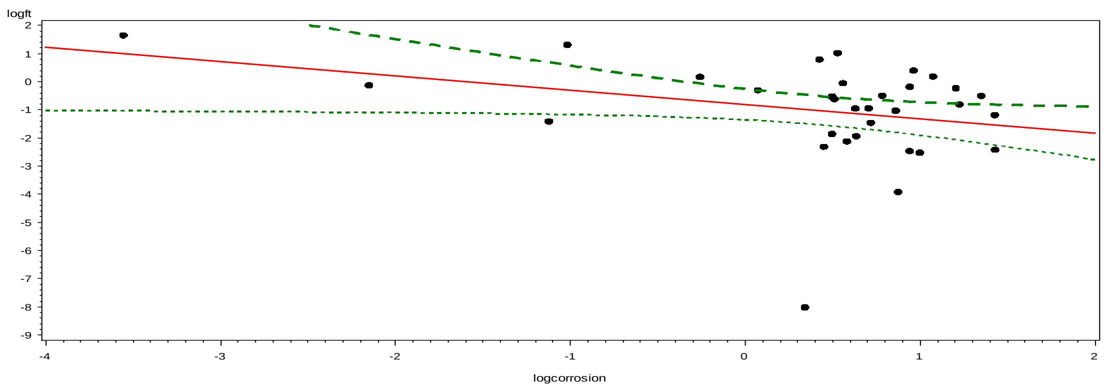
Attempt1: Log(time) VS Corrosion



Attempt2: Time VS Log(Corrosion)



Attempt3: Log(Time) Vs Log(Corrosion)



Since more observed values are inside of prediction region, three transformed linear regression models perform better than the original one. However, there are still many observations that fall out the predication boundaries. Therefore, we should try to use other method to model the parameters.

Various way to model the data

Traditionally, the Weibull distribution is used to model life spans of people or an object. For the sake of simplification, exponential distribution as one particular kind of the Weibull distribution is used in this article. In the exponential distribution, only one parameter (λ) need to be modeled. There are two ways to model λ.

Model1: $\lambda= aw^b$, where w is degree of corrosion
Model2: $\lambda= \exp(-\mu)$, where $\mu=\beta_0+\beta_1 w$

Model1 is an intuitive way to model our parameter. Since it is know that life span will be a positive number ($\lambda>0$) and a univariate model is used, polynomial model come to the sense naturally.

Model2 is accelerated failure time model, which is often used in survival analysis in biostatistics. This model is performed in SAS LIFEREG procedure with the assumption of exponential distribution.

Table 1 shows the analysis of MLE estimates and model log-likelihood

Model	Estimate A	Estimate B	CI for B	Model log-likelihood
Model1	1.133005(a)	0.4792074(b)	(0.1338348, 0.8245801)	-21.71021
Model2	0.6184(intercept)	-0.4503(corrosion)	(-0.7416, -0.1590)	-55.3984948

Even through two models are under the same distribution, the parameter estimates and likelihood are very different. It makes people wandering how to connect two models. Therefore, model3 is created to explore the connection.

Model3 is performed under SAS LIFEREG procedure, using -log(corrosion) as factor.

Table 2

Model	Estimate A	Estimate B	CI for B	Model log-likelihood
Model1	1.133005(a)	0.4792074(b)	(0.1338348, 0.8245801)	-21.71021
Model3	-0.1249 (intercept)	0.4792 (-log(corrosion))	(0.1338, 0.8246)	-54.37371518

Notice that Model3 has same MLE and CI for estimate B. Also, if you take $e^{-0.1249}$ in estimate A of Model 3, it will equal to 1.133005 in estimate A of Model1. The difference between two models will be model log-likelihood.

Model4 is produced to fix the log-likelihood. Model4 is modified from Model1 with the same parameter setup but transforming failure time to log(failure time).

Table 3

Model	Estimate A	Estimate B	CI for B	Model log-likelihood
Model3	-0.1249 (intercept)	0.4792 (-log(corrosion))	(0.1338, 0.8246)	-54.37371518
Model4	1.133005(a)	0.4792074(b)	(0.1338348, 0.8245801)	-54.37372

Model3 and Model4 are mathematically "close", since they have the same MLE, CI for B and log-likelihood.

Mathematical connection between models

Let's start with equations to model λ. Model3 changes the factor in Model2 from corrosion to -log(corrosion). Mathematically,

$$\begin{aligned}\lambda &= \exp(-\beta_0 - \beta_1 * (-\log(w))) \\ &= \exp(-\beta_0) * \exp(+\beta_1 * \log(w)) \\ &= \exp(-\beta_0) * \exp(+\log(w)^{\beta 1}) \\ &= \exp(-\beta_0) * w^{\beta 1}, \qquad (1)\end{aligned}$$

where w is the corrosion, β_0 is intercept, β_1 is estimate B.

If we set $a = \exp(-\beta_0)$, the equation of λ in Model3 will have the same form with Model1, where $\lambda = aw^b$.

Then, Model4 log transforms failure time and matches with the log-likelihood in Model3. Mathematically, in model1

X1, X2,…… , Xn follows Exp(λ)

$f(x) = \lambda\exp(-\lambda x)$

Also, we set $\lambda = aw^b$

Then, the loglikehood,

$$L(a, b) = n * \log(a) + b * \sum_1^n \log(wi) - a * \sum_1^n wi^b * ft \qquad (2)$$

Third step, since we know the two models parameterization is essentially the same; the only difference will only come from the transformed response.

In Model3, failure time is transformed. Mathematically,

again, X1, X2,….., Xn comes from Exp(λ)
$f(x)= \lambda \exp(-\lambda x)$

Set Y= Ln(X), the logarithm transformation.

$f(y)= f(e^y) * |de^y / dy|$, by proposition D

$$f(y)= \lambda * \exp(-\lambda e^y) * e^y$$
$$=\lambda * \exp(y-\lambda e^y) \qquad (3)$$

Plug in $\lambda= aw^b$ into (3)

$$f(y)= aw^b \exp(y- aw^b e^y) \qquad (4)$$

The likelihood from (4)

$$L(a,b)= f(x1,x2,\ldots..xn| a, b)= \prod_1^n [aw^b \exp(y - aw^b e^y)] \qquad (5)$$

The log-likelihood from (5)

$$ll(a,b)= \log\{\prod_1^n [aw^b \exp(y - aw^b e^y)]\}$$
$$=\sum_1^n \log(aw^b \exp(y - aw^b e^y))$$
$$= \sum_1^n [\log(a) + b * \log(wi) + yi - awi^b e^{yi}]$$
$$= n*\log(a)+b*\sum_1^n \log(wi)+\sum_1^n yi- a*\sum_1^n wi^b e^{yi} \qquad (6)$$

Notice that, yi is log(failure time) so that e^{yi} = failure time, which is ft in the R program. Comparing with (1) and (6), the only difference between logarithm transformed log-likelihood and original log-likelihood is $\sum_1^n yi$.

Discussion

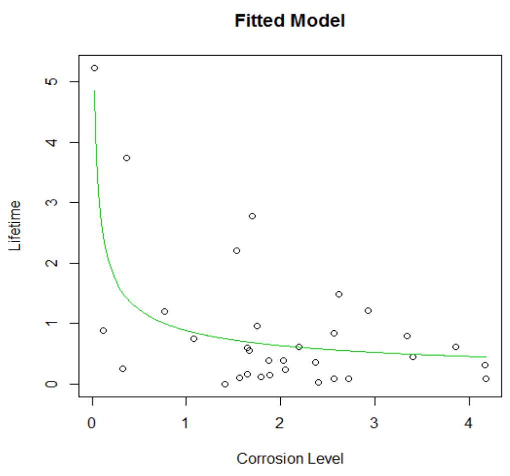
1. The accelerated failure time model can be reduced to a more intuitive model, such as model4, by setting covariates to negative log transformed covariates. This model is not necessarily meaningful in the practice but it provides unique insight on the mathematics behind the model and relationship between different models under the same assumption.

2. Logarithm transformation will move the model log-likelihood up or down in parallel. The deviance which is the distance between null hypothesis and alternative hypothesis will not change.

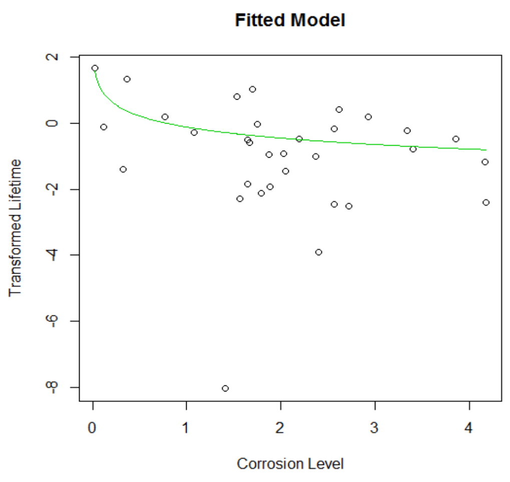
3. After log transformation, MLEs of parameters in the model don't change. Also, the standard errors and confidence interval of those covariates don't change as well. This property means the transformation won't change the relation between response and factors, therefore let us to do the log transformation freely without making arbitrarily changes in the dataset.

4. The log transformation does not necessarily improve the model fit. In fact, in the case Model 1 and Model4, original model performs better than the transformed model, due to outliers in Model4.

Model 1



Model4



Appendix

1. Modified R code from Dr. Suess's ft.R code.

```
library(ismev)
library(stats4)

# data from the ismev library
#

data(engine)

ft = engine$Time        # failure time
n = length(ft)          # sample size
w = engine$Corrosion    # corrosion measurement

plot(w,ft)

log.ft=log(ft)
log.w=log(w)

# MLE

# solve the nonlinear equation

# minus the log likelihood

ll = function(a,b) {
   -n*log(a) - b*sum(log(w))- sum(log.ft) + a*sum((w^b)*exp(log.ft))
}

model.mle = mle(minuslog=ll,start=list(a=1,b=1))
model.mle

a.mle = coef(model.mle)[1]
a.mle
b.mle = coef(model.mle)[2]
b.mle

ll.value = -ll(a.mle,b.mle)
ll.value

# CIs for a and b, the CI for b matches with accelerated ft model
# of transformed corrison level.
```

```
I.O = matrix(c( n*a.mle^-2, sum(w^b.mle*ft*log(w)),
sum(w^b.mle*ft*log(w)), a.mle*sum(w^b.mle*ft*log(w)^2) ), c(2,2))
I.O

I.O.inv = solve(I.O)    # produces the matrix inverse
I.O.inv

a.mle.se = sqrt(I.O.inv[1,1])
a.mle.se

conf.level = 0.95
cv = qnorm(1-(1-conf.level)/2)

a.ci = c(a.mle - cv*a.mle.se,a.mle + cv*a.mle.se)
a.ci


b.mle

b.mle.se = sqrt(I.O.inv[2,2])
b.mle.se

b.ci = c(b.mle - cv*b.mle.se,b.mle + cv*b.mle.se)
b.ci


# Test if H0: b=0, H1: b=/= 0
a.mle.small = n/sum(ft)
a.mle.small

# log-likelihood of null hypothysis
ll.value.small = n*log(a.mle.small)+sum(log.ft)-a.mle.small*sum(exp(log.ft))
ll.value.small

# Deviance is the same with the model before the log transformation
D = 2*(ll.value - ll.value.small)
D


# plot fitted model, I am not sure this is right.

a.mle = coef(model.mle)[1]
a.mle
```

```
b.mle = coef(model.mle)[2]
b.mle

w.index = seq(min(w),max(w),0.01)

ft.fit = (a.mle^-1)*w.index^(-b.mle)
log.ft.fit=log(ft.fit)

plot(w,log.ft,xlab="Corrosion Level",ylab="Transformed Lifetime",main="Fitted
Model")
lines(w.index,log.ft.fit,type="l",col=3)
```

2. SAS code

```
data mydata;
input time 12.10 corrosion 12.10 ;
datalines;
  5.231237563 0.02856561
  0.883741162 0.11644553
  0.245824519 0.32556412
  3.737508046 0.36187570
  1.193548683 0.77289500
  0.744449009 1.07671243
  0.000331672 1.40806603
  2.212633058 1.53019431
  0.099889341 1.56819203
  0.157013076 1.64420582
  0.593876487 1.64440864
  0.545076312 1.66461209
  2.782713173 1.69701454
  0.955511842 1.74957354
  0.120548481 1.78876443
  0.388568088 1.87775873
  0.145561389 1.88814442
  0.392746324 2.02600741
  0.234012534 2.05149663
  0.613340116 2.19011591
  0.359726135 2.36558148
  0.020013325 2.39948193
  0.085350498 2.56172240
  0.837877708 2.56528502
  1.491809687 2.62078743
  0.080670417 2.71643983
  1.210000996 2.92964335
  0.798518117 3.33795520
  0.450192367 3.40658882
  0.609792042 3.86109929
  0.308168774 4.16830998
  0.089619767 4.17895697
;
run;

data mydata1;
set mydata;
logcorrosion=-log(corrosion);
logcorrosion1=log(corrosion)
logft=log(time);
run;

symbol2 v=dot i=rlclm90 width=2 cv=black ci=red co=green;
proc gplot data=mydata1;
plot time*corrosion;
plot logft*corrosion;
plot time*logcorrosion1;
plot logft*logcorrosion1;
run;

proc lifereg data=mydata1;
model time= corrosion/ dist=exponential covb;
run;

proc lifereg data=mydata1;
model time= logcorrosion/ dist=exponential covb;
output out= stat p=predict std=stde xbeta=phi;
run;

proc print data=stat;
run;



quit;
```