

Projecte Sobre Weibull de Simulació

Arnau Pérez Reverte, Pau Soler Valadés

04-12-2025

Simulació, MESIO UPC-UB

1. Introducció

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua quaerat voluptatem. Ut enim aequaleam animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut.

2. Objectiu (Aims)

L'objectiu principal del següent estudi de simulació és l'estudi de la inferència dels paràmetres d'una distribució de Weibull mitjançant el mètode Median Ranks Regression (MRR). Amb aquesta finalitat, es desenvoluparà la teoria al voltant d'aquesta distribució de probabilitat, es definirà aquest procediment d'inferència en detall i s'avaluarà la qualitat dels seus resultats mitjançant una sèrie de mètriques, així com es compararà amb el mètode ordinari de inferència del Maximum Likelihood Estimation (MLE).

3. Generació de Dades (Data-generating mechanisms)

3.1. La distribució de Weibull

Diem que una variable aleatòria contínua X segueix una distribució de Weibull de paràmetres α i β , $X \sim \text{Weibull}(\alpha, \beta)$, si té funció de densitat de probabilitat (pdf)

$$f_X(x; \alpha, \beta) = \left(\frac{\beta}{\alpha}\right) \left(\frac{x}{\alpha}\right)^{\beta-1} \exp\left(-\left(\frac{x}{\alpha}\right)^\beta\right) 1_{\{x>0\}},$$

on $\alpha > 0$ i $\beta > 0$ són els paràmetres scale i shape, respectivament. S'observa aleshores que la seva funció de distribució de probabilitat (cdf) pren la forma

$$F_X(x) = 1 - \exp\left(-\left(\frac{x}{\alpha}\right)^\beta\right).$$

La distribució de Weibull, amb els seu suport als nombres reals positius, resulta a la pràctica molt útil per modelitzar el concepte de "temps de vida" en contextos com control de qualitat, epidemiologia, entre d'altres. Els seus paràmetres a més a més es poden, en aquesta idea mateixa idea, caracteritzar i interpretar de la forma següent:

- α (Scale): També conegut com vida característica, pren unitats d'acord al context del problema, és a dir, ja sigui perquè estem modelitzant segons, minuts, hores, cicles, etc.
- β (Shape): En el nostre context, β és un paràmetre adimensional conegut com la proporció de fallada, i que modifica el comportament de la distribució de Weibull de la següent forma en funció dels tres casos:
 1. $\beta < 1$: Proporció de fallada decreixent/"mortalitat infantil". Com major és el temps de supervivència de la unitat, menor és la probabilitat de que mori/falli en el següent instant.
 2. $\beta = 1$: Proporció de fallada constant. Simplifica la distribució de Weibull a una $\text{Exp}(\frac{1}{\alpha})$, simbolitzant que el temps de supervivència és aleatori i independent de temps transcorregut.

3. $\beta > 1$: Proporció de fallada creixent. La probabilitat de mort creix a mesura que el temps incrementa. Representa unitats/individus que empitjoren amb el pas del temps.

Com ja s'ha descrit, la distribució de Weibull es s'utilitza en el contexte del temps i en conseqüència manté una relació molt estreta amb altres distribucions utilitzades amb aquesta mateixa idea. Es destaca, en particular: (TODO REVISAR)

1. $\text{Weibull}(\alpha, 1) = \text{Exp}\left(\frac{1}{\alpha}\right)$
2. $X \sim \text{Weibull}(\sqrt{2}\beta, 2) = \text{Rayleigh}(\beta)$

3.2. Mètode de generació de nombres aleatòries

Per tal de simular valors de la distribució de Weibull, s'utilitzarà el mètode de distribució inversa. En efecte, el mètode emprà la composició de dues funcions: la funció de densitat d'una distribució uniforme $U(0, 1)$ juntament amb la inversa de la funció de probabilitat F de la distribució que es vulgui generar. Una variable aleatòria $U \sim U(0, 1)$ té la densitat f_U següent:

$$f_U(x) = 1I_{\{0 \leq x \leq 1\}}(x)$$

i està definida en $f_U : \mathbb{R} \mapsto [0, 1]$, i una CDF seguint una variable aleatòria X està definida a $F_X : \mathbb{R} \mapsto [0, 1]$, i la seva inversa per tant a $F_X^{-1} : [0, 1] \mapsto \mathbb{R}$. Aleshores, la seva composició

$$F_X^{-1} \circ f_U \sim X$$

$$F_X^{-1} \circ f_U : \mathbb{R} \xrightarrow{f_U} [0, 1] \xrightarrow{F_X^{-1}} \mathbb{R}$$

No només és possible, sinó que és una CDF de la variable X . Això és possible ja que U genera els nombres uniformement entre l'interval $[0, 1]$, fent que la imatge de la composició es comporti com la distribució de la variable aleatòria caldria esperar.

En el cas de la distribució de Weibull, és fàcil veure que la funció inversa de la CDF està ben definida i pren la forma:

$$F_X^{-1}(u) = \alpha(-\log(u))^{\frac{1}{\beta}}.$$

3.3. Implementació computacional

La implementació de les simulacions es realitza en el marc del llenguatge de programació Python utilitzant principalment les llibreries de computació numèrica NumPy i SciPy (TODO citar), les quals proporcionen una API d'alt nivell fàcil d'utilitzar, amb implementacions numèriques eficients testejades per la comunitat. Els objectes numèrics que s'utilitzaran són principalment del tipus `np.array`, permetent operacions vectoritzades que acceleren la realització de simulacions.

Un dels focus d'aquest treball i especialment de l'assignatura és la garantia de reproducibilitat d'un estudi de simulació, és a dir, desenvolupar el codi de la simulació de tal forma que qualsevol usuari interessat en reproduir (de forma exacta) els fets presentats en aquesta memòria ho pugui fer sense problema.

Amb aquest fi, el codi annex a la memòria s'estructura fonamentalment en els següents blocs de codi:

1. En primer lloc, el mòdul `src/dgm.py` conté la classe `WeibullDistribution` la qual inicialitzada amb els paràmetres `alpha` i `beta` adequats encapsula una $\text{Weibull}(\alpha, \beta)$ de la qual es pot generar una mostra de mida n mitjançant el mètode `sample(n: int)`, que implementa el mètode descrit a l'apartat anterior. La generació de nombres aleatoris ben definida la proporciona NumPy amb `np.random.default_rng(seed)`, que per darrera implementa l'algoritme de Mersenne-Twister proporcionada una llavor (`seed`). S'observa que, creat un objecte generador de nombres aleatoris de NumPy, la classe `WeibullDistribution` permet ser inicialitzada amb aquest específicament, de forma que si es canvia de paràmetres durant

l'estudi de simulació, el fet de compartir el generador garanteix el control total sobre la seqüència de nombres que es van generant.

2.

4. Estimand

Volem estimar els valors d' α, β de la weibull empíricament, donades les dades que hem generat per simulació.

Per a cada combinació dels paràmetres (n, α, β) generarem m repetitions independents. Per a cadascuna d'aquestes, calcularem els estimadors d' α, β tan amb el mètode MLE $(\widehat{\alpha}_{MLE}, \widehat{\beta}_{MLE})$ i amb el MLE $(\widetilde{\alpha}_{MLE}, \widetilde{\beta}_{MLE})$ obtenint una distribució empírica dels estimadors per anàlitzar-ne les mètriques llistades a l'apartat dels resultats.

5. Performance Measures

Les mesures que utilitzarem seran:

1. Biaix: $\text{Biaix} = \frac{1}{m} \sum_{i=1}^m (\hat{\theta}_i - \theta)$
2. Error estàndard de montecarlo (MCSE): $\text{MSCE}(\text{Biaix}) = \sqrt{\frac{1}{m(m-1)} \sum_{i=1}^m \hat{\theta}_i - \bar{\hat{\theta}}}$
3. MSE: $\frac{1}{m} (\hat{\theta}_i - \theta)^2$

Per a comparar-lo amb l'MLE, calcularem les mètriques anteriors també per l'MLE i les contrastarem amb les del MRR.

6. Results

Gràfiques: per a cada tamany mostral, posem les gràfiques que estan a `plotting.py`: Taules de resultats: TODO: La taula hauria de ser

1. files: n
2. Columnes: Biax, MCSE MSE de MRR i després de MLE
3. Hi haurà 9 taules, una per cada valor diferent de alpha beta.

7. Annex

Posar totes les classes a lo copia i enganxa. Més enllà, fer ènfasi al main, i fer un esforç en que imprimieixi els mateixos resultats exactes que es presenten al report. (EG la taula, s'ha de trobar com fer-ho)