

# Estimació dels paràmetres de la distribució Weibull

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. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. 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)

El DGM defineix com s'utilitza el mostreig pseudoaleatori per crear dades. Aquest estudi contindrà exclusivament simulacions paramètriques de dades distribuïdes seguint una Weibull per a diferents configuracions dels paràmetres  $(\alpha, \beta)$  per tal d'experimentar amb l'estimació en funció de la forma de la distribució, així com de la mida de la mostra.

Fixats valors de  $n > 0$  i  $\alpha, \beta > 0$ , es defineix que el DGM que es realitza en una única simulació consisteix en la generació d'una mostra d'observacions  $T_1, T_2, \dots, T_n$  i.i.d, amb  $T_i \sim \text{Weibull}(\alpha, \beta)$ . Els factors a variar per l'experimentació del mètode a estudiar seran aleshores  $(\alpha, \beta, n)$ , on seguint l'enunciat de la pràctica es defineix la següent matriu de valors basats en tres nivells (*small, medium, large*):

Factor	Tipus	Valor		
		<i>Small</i>	<i>Medium</i>	<i>Large</i>
$\alpha$ (Scale)	Paràmetre de distribució	2	10	50
$\beta$ (Shape)	Paràmetre de distribució	0.8	1	3
$n$ (Mida de la mostra)	Paràmetre de mostreig	10	50	200

Figura 1: Matriu de factors a variar a l'estudi

Els valors proposats pels tres nivells de la mida de la mostra  $n$  són valors estàndards dintre de la literatura. En el cas dels paràmetres  $\alpha, \beta$ , primer es definiran amb detall a continuació i aquesta proposta de valors quedarà aleshores justificada.

### 3.1. La distribució de Weibull

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

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

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 al 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:

- $\alpha$  (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.
- $\beta$  (Shape): En el nostre context,  $\beta$  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 del 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 s'utilitza en el context del temps i en conseqüència manté una relació molt estreta amb altres distribucions utilitzades amb aquesta mateixa idea. Es destaca, en particular:

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

### 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 empra 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:

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

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 per  $u \in (0, 1]$  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 [1] i SciPy [2], 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 cap inconvenient. Amb aquest fi, el codi annex a la memòria s'estructura fonamentalment en diferents blocs de codi.

En primer lloc, el mòdul `src/dgm.py` conté la classe `WeibullDistribution` la qual inicialitzada amb els paràmetres  $\alpha$  i  $\beta$  adequats encapsula una  $Weibull(\alpha, \beta)$  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'algorisme de Mersenne-Twister proporcionada una llavor (*seed*). S'observa que, creat un objecte generador de nombres aleatoris de NumPy `rng`, la classe `WeibullDistribution` permet ser inicialitzada amb aquest generador 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 a cada punt del programa.

## 4. Mètodes

TODO

### 4.1. Mètode 1: Maximum Likelihood Estimation (MLE)

TODO

### 4.2. Mètode 2: Median Ranks Regression (MRR)

El mètode de Median Ranks Regression (MRR) per a l'estimació dels paràmetres consisteix en tres passos:

1. Donada la mostra d'observacions de temps de fallada  $T_1, \dots, T_n$ , primer s'ordena la mostra en ordre creixent, obtenint així la mostra  $\hat{T}_1, \dots, \hat{T}_n$  on

$$\hat{T}_i := T_{(i)}.$$

Es guarda també la permutació  $\sigma : \{1, \dots, n\} \rightarrow \{1, \dots, n\}$  tal que  $\sigma(T_1, \dots, T_n) = (\hat{T}_1, \dots, \hat{T}_n)$ .

2. Es calcula el rang medià  $MR_j$  de cada fallada  $\hat{T}_j$  de la mostra mitjançant l'equació

$$0.50 = \sum_{k=j}^N \binom{N}{k} MR_j^k (1 - MR_j)^{N-k}$$

i s'utilitza com a estimació de la no-fiabilitat:  $Q(\hat{T}_j) \approx MR_j$ .

Aquesta equació es pot resoldre en primera instància utilitzant algorismes de cerca d'arrels com el mètode de Newton. No obstant, es pot veure a [3, p. 15] que el costat de la dreta coincideix amb el valor  $I_{MR_j}(j, n - j + 1)$ , que es correspon amb l'anomenada funció beta incompleta regularitzada, definida com

$$I_x(a, b) = \frac{B(x; a, b)}{B(a, b)},$$

on  $B$  és la funció Beta incompleta

$$B(x; a, b) = \int_0^x t^{a-1}(1-t)^{b-1} dt$$

i posem  $B(a, b) := B(1; a, b)$ . Per tant,

$$0.50 = I_{MR_j}(j, n - j + 1)$$

i el problema es simplifica a treballar amb la inversa (sobre  $x$ ) de  $I_x(a, b)$ , la qual es troba implementada en paquets de programari estadístic [2].

És fàcil observar a més a més que aquest càlcul del  $MR_j$  depèn només de  $n$  i de  $j$ , i no del valor observat de  $\hat{T}_j$ , el qual s'incorporarà més endavant. És per això que una molt bona aproximació al valor real de l' $MR_j$  es pot obtenir mitjançant

$$MR_j \approx \frac{j - 0.3}{n + 0.4}.$$

3. Es transformen les estimacions de no-fiabilitat mitjançant l'aplicació

$$Q \mapsto \log(-\log(1 - Q)).$$

Aquesta funció està ben definida, ja que  $Q \in (0, 1)$  i  $-\log(1 - Q) > 0$ . S'observa que la funció transforma la *cdf* de la Weibull en una recta. En efecte,

$$Q := F_X(x; \alpha, \beta) = 1 - \exp\left(-\left(\frac{x}{\alpha}\right)^\beta\right)$$

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

$$-\log(1 - Q) = \left(\frac{x}{\alpha}\right)^\beta$$

$$\log(-\log(1 - Q)) = \beta \log(x) - \beta \log(\alpha).$$

4. Amb la mostra de parells de temps de fallada i les seves estimacions de no-fiabilitat  $(T_{\sigma^{-1}(j)}, Q_j)$ , s'utilitza Mínims Quadrats per ajustar una recta i extreure els coeficients resultants  $m$  i  $b$ . Llavors, la inferència sobre els paràmetres del nostre model de Weibull es pot aconseguir mitjançant

$$m \equiv \beta$$

$$b \equiv -\beta \log(\alpha) \Rightarrow \alpha = \exp\left(-\frac{b}{\beta}\right).$$

### 4.3. Implementació computacional

Els diferents mètodes d'estimació descrits prèviament s'implementen com a funcions al mòdul `src/inference.py`.

1. **Median Ranks Regression (MRR):** En el cas del MRR, la funció `median_ranks_regression` pren una mostra de simulació arbitrària i implementa el procediment explicat anteriorment. En particular, el paràmetre `method` permet escollir entre "beta" i "bernard", per calcular els rangs medians utilitzan la forma explícita de la funció

beta incompleta regularitzada (disponible per `from scipy.special import betaincinv`) o bé l'aproximació de Bernard. En ambdós casos s'utilitzen operacions matricials de NumPy, de forma que el procés és vectoritzat i per tant més ràpid. Finalment, els Mínims Quadrats es computen mitjançant `linregress` de `scipy.stats`.

## 2. Maximum Likelihood Estimation (MLE): TODO

## 5. Estimand

Volem estimar els valors d' $\alpha, \beta$  de la weibull empíricament, donades les dades que hem generat per simulació.

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

## 6. 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.

## 7. Results

Resultats: he agrupat en csv per alpha beta. Cadascun conté 3 files, una per a cada n. proposo escriure directament aquestes taules a l'apartat de resultats, partint-ho per MLE, MRR i Betrand

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.

Grafiques (sempre per parella alpha-beta encara que no es digui el contari):

1. Primera: MSE per alpha beta (y-axis) i tipus (x-axis) MLE MRR Bertanrd i MRR normal code. Amb MCSE.
2. Gràfic de línies mse-bias. x: mse, y: bias. tres punts, un per sample size, tres línies, una per mètode.
3. Gràfic de línies. y-axis: mse; x-axis sample-size.

Grafiques sobre bootstrap:

1. boxplot: alph

Les gràfiques mínimes han de ser la regressió lineal amb els mínims quadrats per ajustar una recta. Gràfiques: per a cada tamany mostral, posem les gràfiques que estan a `plotting.py`:  
Taules de resultats:

## 8. Annex

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

## Bibliografia

- [1] C. R. Harris *et al.*, “Array programming with NumPy,” *Nature*, vol. 585, no. 7825, pp. 357–362, Sept. 2020, doi: 10.1038/s41586-020-2649-2.
- [2] P. Virtanen *et al.*, “SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python,” *Nature Methods*, vol. 17, pp. 261–272, 2020, doi: 10.1038/s41592-019-0686-2.
- [3] J. Dutka, “The incomplete beta function – a historical profile,” *Archive for History of Exact Sciences*, vol. 24, no. 1, pp. 11–29, 1981, doi: 10.1007/BF00327713.