ÉCOLE UNIVERSITAIRE DE PHYSIQUE ET D'INGÉNIERIE
Université Clermont Auvergne

2024-2025
M2PFA UP & NP
IMMAP
DU Data Scientist

Statistics homework
October 2024

About the Random Sampling

It is sometimes useful to generate a sample of random variables following an experimental distribution, that may (or not) have an explicit analytical definition. The purpose of this exercise is to explore different techniques used to produce such random samples.

I - Using the reciprocal of the cumulative distribution.

Let's assume that the cumulative function of the target distribution, $\mathscr{F}_{\mathrm{X}}$, is known or, at least, that it can be numerically derived from the experimental data. To generate a random sample of independent random variables X, identically distributed according to the density function with cumulative $\mathscr{F}_{\mathrm{X}}$, we can use a transformation of the continuous *uniform* distribution in the range $[0, 1]$, $\mathrm{U} \sim \mathbb{I}_{[0,1]}$, that has the property to coincide with its own reciprocal: $\mathbf{P}(\mathrm{U} < p) = \mathscr{F}_{\mathrm{U}}(p) = p = \mathscr{F}_{\mathrm{U}}^{-1}(p)$. The *inverse sampling method* is based on the reciprocal theorem that ensures that given the reciprocal function $Q_{\mathrm{X}}(p) = \mathscr{F}_{\mathrm{X}}^{-1}(\boldsymbol{x})$, the random variable defined as $\mathrm{X} = Q_{\mathrm{X}}(\mathrm{U})$ has $\mathscr{F}_{\mathrm{X}}(\boldsymbol{x})$ for cumulative .

For the purpose of this exercise, let's consider for instance the *sigmoid* function, defined as:

$$\mathscr{F}_{\mathrm{X}}(x; \lambda) = \frac{1}{1 + e^{-\lambda x}}.$$

with parameter $\lambda \in \mathbb{R}^+$. This monotically increasing function, $\mathscr{F}_{\mathrm{X}} : x \in \mathbb{R} \rightarrow \mathscr{F}_{\mathrm{X}}(x, \lambda) \in [0, 1]$ is a valid and normalized *cumulative distribution function* (CDF) for a continuous random variable X. It's easy to derive the corresponding reciprocal function, i.e. the *percent point (or quantile) function* (PPF) of the distribution:

$$\mathscr{F}_{\mathrm{X}}(x, \lambda) = \mathbf{P}(\mathrm{X} < x) = p \implies x = Q_{\mathrm{X}}(p, \lambda) = \mathscr{F}_{\mathrm{X}}^{-1}(p, \lambda) = \frac{1}{\lambda} \ln\left(\frac{p}{1-p}\right).$$

1- Generate a sample of $N = 10^5$ random probabilities p_k, uniformly distributed in $[0, 1]$. [1] For each random value p_k, evaluate the corresponding value $x_k = \mathscr{F}_{\mathrm{X}}^{-1}(p_k; \lambda = 1) = \ln(p_k/(1-p_k))$. Draw the density histogram of the sample $\{x_k\}$.
You now have a random sample of values $\mathrm{X} \sim f_{\mathrm{X}}(x; \lambda = 1)$, for the parameter value $\lambda = 1$.

2- It is rather easy to derive the explicit expression of the *probability density function* (PDF):

$$f_{\mathrm{X}}(x, \lambda) = \frac{\partial \mathscr{F}_{\mathrm{X}}(\boldsymbol{x}; \lambda)}{\partial x} = \frac{\lambda}{4} \operatorname{sech}^2\left(\frac{\lambda x}{2}\right),$$

where $\operatorname{sech}(z) = 1/\cosh(z)$ is the *hyperbolic secant.*
Superimpose the $f_{\mathrm{X}}(x, 1)$ density curve on the sample histogram. Does it match ?

[1] Adapt N to your computing ressources. You can start with a lower value to develop your code, N=10^4 for instance, and increase the sample size as long as the processing time remains reasonable.

3- To further validate the generated distribution, let's estimate the λ parameter of the theoretical PDF using the Maximum Likelihood (ML) method. Define the negative log-likelihood function of the parameter λ, evaluated on your sample $\{x_k\}$:

$$-\ln\mathscr{L}\left(\lambda;\{x_k\}\right) = -\sum_{k=1}^{N} \ln\left[f_{\rm X}(x_k;\lambda)\right],$$

and extract the ML estimator $\hat{\lambda}$ that minimizes this function.

4- Evaluate the corresponding uncertainty, $\hat{\sigma}_\lambda$, using the graphical $\Delta\ln\mathscr{L}(\lambda_\pm) = 1/2$ method. Is the value measured on the sample, $\hat{\lambda}\pm\hat{\sigma}_\lambda$, consistent with the expected value of the theoretical PDF, $\lambda = 1$?

5- If yes, congratulations! You can now generate a random sample from any CDF.[2] However, you have worked for nothing but the glory, because this sech-squared PDF is well-known and already implemented in the *scipy* library under the name of *logistic distribution*, with all the usual machinery for random generation, evaluation of the PDF, CDF, PPF, log-Likelihood ... Produce an alternative sample of N random values according to the *logistic* density with the *scipy* generator (scipy.stats.logistic.rvs(size=N)). Compare this sample distribution with your own sampling generation, by plotting the *residuals histogram*, i.e. the histogram of the difference in each *bin* of the two histograms, divided by the associated uncertainty. You can approximate the uncertainty on the *bin* content with $\sigma_{n_i} \simeq \sqrt{n_i}$ (be careful with the density normalisation). Conclusion.

6- The *logistic* density exhibits a symmetrical profile with fatter tails than the Gaussian-Normal distribution (these kind of profiles are known as *leptokurtic*). The Expectation and the Variance of the *logistic* distribution, $\mathrm{X} \sim f_{\rm X}(x,\lambda)$, are:

$$\mathbb{E}\left[\mathrm{x}\right] = 0 \quad \text{and} \quad \mathbb{V}\left[\mathrm{x}\right] = \frac{\pi^2}{3\lambda^2}.$$

Let's try to play with this known Variance to evaluate the number π.
Use the random sample $\{x_k\}$, generated with the parameter value $\lambda = 1$, to build a new sample $\{z_k\}$, where the random variable Z is the standardized X value scaled by the factor $1/\sqrt{3}$, i.e.

$$z_k = \frac{x_k - \bar{\mu}_{\rm X}}{\sqrt{3\bar{\sigma}^2_{\rm X}}},$$

with $\bar{\mu}_{\rm X}$ and $\bar{\sigma}^2_{\rm X}$, the mean and the variance of the sample $\{x_k\}$, respectively.[3] In the limit of large N, the sample variance approximates the distribution Variance, $\bar{\sigma}^2_{\rm X} \simeq \mathbb{V}\left[\mathrm{x}\right]$. The Variance of the rescaled variable Z is then $\mathbb{V}\left[\mathrm{z}\right] \simeq 1/3$ and its PDF is a *logistic* distribution, $f_{\rm Z}(z;\lambda_{\rm Z})$ with $\lambda_{\rm Z} = \pi$.
Repeat questions I-3 and I-4 to evaluate the ML estimator $\hat{\lambda}_{\rm Z}$ that minimizes the negative log-Likelihood function, $-\ln\mathscr{L}\left(\lambda;\{z_k\}\right)$, evaluated on the rescaled sample $\{z_k\}$, and the associated uncertainty.[4] Is it compatible with the known π value ? (you can also conclude that sampling methods are highly inefficient to determine a large number of digits of π !)

[2]In case the analytical quantile function PPF cannot be derived easily from the experimental CDF, you can always interpolate numerically the latter to extract the x values corresponding to any $p = \mathscr{F}_{\rm X}(x)$ value.

[3]You can get it quickly by importing the *statistics* module: $\bar{\mu}_{\rm X}$ = statistics.mean(X) and $\bar{\sigma}^2_{\rm X}$ = statistics.variance(X).

[4]The uncertainty on the determination of $\bar{\sigma}_{\rm X}$ is neglected here.

II - The accept-reject sampling method.

The accept-reject method is a *Monte-Carlo* technique that allows to "sculpt" a random sample for any target PDF, $g_{\mathrm{X}}(x)$, using a known reference density $f_{\mathrm{X}}(x)$ that verifies the criteria

$$\alpha f_{\mathrm{X}}(x) \geq g_{\mathrm{X}}(x) \; \forall \; x \in supp\,(g_{\mathrm{X}})\,,$$

where α is a scaling contant that ensures that the inequality holds for all x.
The methods works as follow:

i) generate one random probability p uniformly distributed in $[0, 1]$.

ii) generate one random x value according to the known reference density $f_{\mathrm{X}}(x)$.

iii) if the density ratio $g_{\mathrm{X}}(x)/\alpha f_{\mathrm{X}}(x) \geq p$ accept the value x (else reject it).

iv) repeat the procedure N times

The accept-reject method generally requires large reference samples, in particular to populate the low density regions of the target PDF. The closer the target and the reference shapes, the lower the rejection rate.
Let's try to generate a random sample of Normally-distributed events, using the fatter *logistic* distribution as reference.

1- The maximum of the density function is $f_{\mathrm{X}}(0; 1) = 1/4$ for the *logistic* distribution, and $\mathcal{N}_{\mathrm{X}}(0; 0, 1) = 1/\sqrt{2\pi}$ for the *Normal* distribution. We can choose the minimal scaling factor as $\alpha_0 = 4/\sqrt{2\pi}$ to ensure the *logistic* distribution wrap the *Normal* one for all x values.
Draw the two functions, $\alpha_0 f_{\mathrm{X}}(x, 1)$ and $\mathcal{N}_{\mathrm{X}}(x; 0, 1)$, on the same graph to verify that we have $\alpha_0 f_{\mathrm{X}}(x) \geq \mathcal{N}_{\mathrm{X}}(x) \; \forall \; x$.[5]

2- For each event in the random *logistic* sample previously generated $\{x_k\}$, $(k = 1, N)$
(i) evaluate the ratio $r(x_k) = \mathcal{N}_{\mathrm{X}}(x_k)/\alpha_0 f_{\mathrm{X}}(x_k, 1)$.
(ii) generate a random probability p uniformly distributed in $[0, 1]$
(iii) if $r(x_k) \geq p$ save x_k in the subsample of accepted values $\{x_m\}_a$, $m = 1, M \leq N$.
What is the acceptance rate M/N ?

3- Draw the density histogram for the subsample $\{x_m\}_a$. Verify it exhibits the expected *Normal* profile by performing a fit of the Gaussian PDF to the $\{x_k\}_a$ subsample (use scipy.stats.norm.fit, for instance). Give the obtained $\hat{\mu}$ and $\hat{\sigma}$ fit values and draw the corresponding PDF, $\mathcal{N}(x; \hat{\mu}, \hat{\sigma})$, on the histogram.

4- While usually not the optimal choice, the *Uniform* distribution in the finite range $[a, b]$, which is wrapping any other distribution when a proper scaling is applied, can be used as the reference distribution to produce any random sample in a finite range.
Determine the optimal scaling factor α_0 and produce a random *logistic* sample for $\mathrm{X} \sim f_{\mathrm{X}}(x; 1)$ in the range $[-10, 10]$ using an *Uniform* distribution as reference. What is the acceptance rate M/N ? Try to find a better reference choice than the Uniform PDF.

[5]The method works with any scaling $\alpha \geq \alpha_0$, but larger value implies higher rejection rate