

# INFORME TAREA XX

Valentino González  
RUT:XX.XXX.XXX-X  
Github: @thevalentino

## 1. Introducción

El objetivo de esta tarea es encontrar el valor de  $a$  que resuelve la Ecuación (1) del enunciado. Si  $x$  es una variable aleatoria sacada de la distribución descrita en el enunciado, entonces  $a$  representa un valor que asegura que los valores aleatorios de  $x$  serán mayores que  $a$  solo un 5% de las veces.

$$0.05 = \int_a^\infty \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{y^2}{2}\right) dy \quad (1)$$

La ecuación a resolver incluye el cálculo de una integral que debe ser regularizada pues sus límites van de 0 a  $\infty$ . Implementaremos un cambio de variable que regulariza los límites y luego una función que calcule la integral de forma numérica para resolverla para distintos valores de  $a$ .

Finalmente implementaremos una función que busque el valor de  $a$  que hace que la integral original valga 0.05.

## 2. Desarrollo

Lo primero es regularizar la integral. Para ello seguimos la sugerencia del enunciado y aplicamos el cambio de variable  $u = 1/y$ , con lo que el problema se transforma en:

$$0.05 = \int_0^{1/a} \frac{1}{\sqrt{2\pi}u^2} \exp\left(\frac{-1}{2u^2}\right) du \quad (2)$$

La forma de la función a integrar se puede apreciar en la Figura 1. Llamaremos a esta función  $f(u)$ .

Para integrar  $f(u)$  utilizaremos la función `scipy.interpolate.quad`, la cual es una implementación de la librería `QUADPACK` escrita en `FORTRAN`. Esta librería implementa el método de *cuadratura adaptativa* e intenta automáticamente escoger el algoritmo necesario para calcular la integral con una precisión dada por el usuario. Para más información: <https://en.wikipedia.org/wiki/QUADPACK>

Definimos entonces la función:

$$I(a) = \int_0^{1/a} \frac{1}{\sqrt{2\pi}u^2} \exp\left(\frac{-1}{2u^2}\right) du \quad (3)$$

cuya forma se muestra en la Figura (2).  $I(a)$  se calcula numéricamente con la función `quad` con sus parámetros por defecto. Los más importantes son las tolerancias absoluta y relativa:  $epsabs=1.49e-08$ ,  $epsrel=1.49e-08$ , y el límite máximo de sub-intervalos para el algoritmo adaptativo:  $limit=50$ .

Es importante notar que la función  $f(u)$  se indefinice en  $u = 0$ . Sin embargo, como se aprecia en la Figura (1), el área bajo la curva en las cercanías de cero pequeña por, lo que podríamos partir la integral en algún valor  $\epsilon \gtrsim 0$ . El algoritmo de `quad` es capaz de manejar esta indefinición

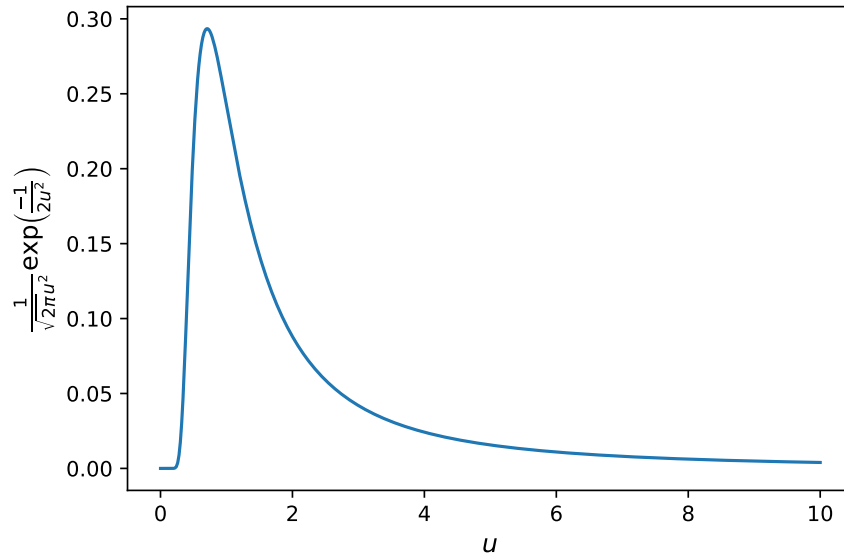


Figura 1: La función que vamos a integrar.

automáticamente, por lo que no implementamos ningún cambio relacionado a la indefinición de la función  $f(u)$ .

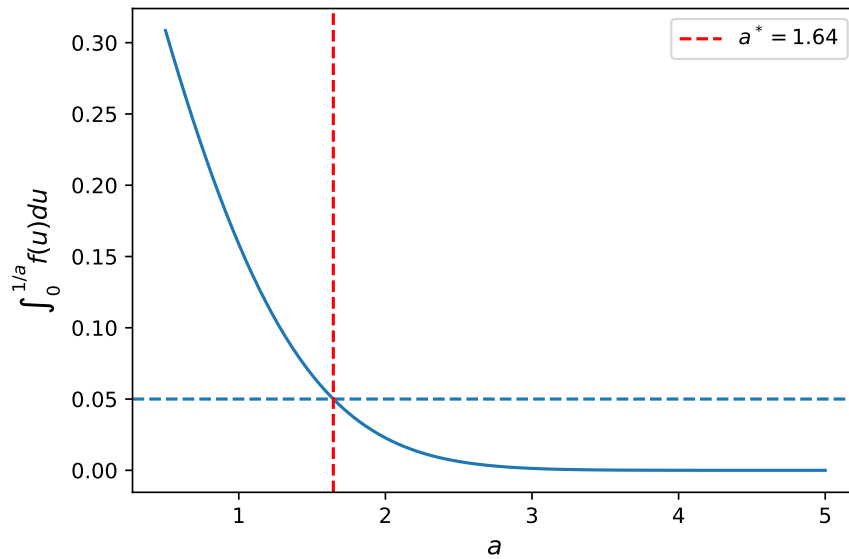


Figura 2: La función  $I(a)$ . Buscamos el valor de  $a$  para el cual la función toma el valor 0.05 (línea azul horizontal punteada). Utilizando el algoritmo de newton, encontramos la solución  $a^* = 1.64$  (línea roja vertical punteada).

Finalmente, es necesario resolver la ecuación:

$$I(a^*) - 0.05 = 0 \quad (4)$$

Esto lo hacemos utilizando el método de newton (en realidad de la secante), implementado en `scipy.optimize.newton`. Utilizamos los parámetros por defecto, en particular, la tolerancia absoluta  $tol=1.48e-08$  y el número máximo de iteraciones  $maxiter=50$ . Le damos como punto de partida el valor  $a_0 = 1$ . El algoritmo converge luego de 7 iteraciones encontrando el valor  $a^* = 1.64485363$ .

Otros valores del punto de partida convergen a un valor muy cercano con diferencias en el número de iteraciones necesarias. Por ejemplo para  $a_0 = 0.5$ , el algoritmo converge luego de 8 iteraciones.

### 3. Discusión y Conclusiones

Si  $x$  es una variable aleatoria sacada de una gausiana con parámetros  $\mu = 0$  y  $\sigma = 1$ , entonces sólo tomará valores mayores a  $a^* = 1.64$  el 5 % de las veces.

Si bien este problema involucra el cálculo de una integral de límites indefinidos (Ecuación 1), el cambio de variable  $u = 1/y$  resulta en una función bien comportada. En particular, la suavidad de la función (ver Figura 2), asegura que algoritmos como el de la *secante* para buscar raíces converjan de manera robusta (casi independiente del punto de partida elegido), y en pocas [sic.] iteraciones.