


Closed-form Expression of Monero's `wallet2`

Decoy Selection Algorithm

Draft v0.1

Rucknium 

October 2023

1 Modified Log-gamma distribution

Let $G(x; \alpha, \beta)$ be the Log-gamma cumulative distribution function (CDF)

$$G(x; \alpha, \beta) = \frac{\gamma(\alpha, \beta \ln(x))}{\Gamma(\alpha)} \quad (1)$$

where α is the shape parameter, β is the rate parameter, γ is the lower incomplete gamma function, and Γ is the gamma function.

In the `wallet2` code, $\alpha = 19.28$ and $\beta = 1.61$. The $G(x; \alpha, \beta)$ is adjusted in the code to:

1. Eliminate the portion of the distribution that would be younger than the youngest spendable output and reallocate it to a uniform distribution in the `RECENT_SPEND_WINDOW`. The $\frac{x/v_t}{1800} \cdot G(1200)$ term of the expression below performs the reallocation.
2. Re-scale the output index distance unit into seconds. The $x \cdot v_t$ term does the re-scaling.
3. Eliminate the portion of the distribution that would be older than the oldest spendable output and reallocate it to the rest of the distribution. Terms with z_t perform this reallocation.

$G^*(x; \alpha, \beta, v_t, z_t)$ is the modified CDF that `wallet2` uses to randomly generate an output index:

$$G^*(x; \alpha, \beta, v_t, z_t) = \begin{cases} 0 & \text{if } x \cdot v_t < 0 \\ (G(x \cdot v_t + 1200) - G(1200) + \frac{x \cdot v_t}{1800} \cdot G(1200)) / G(z_t \cdot v_t) & \text{if } 0 \leq x \cdot v_t \leq 1800 \\ G(x \cdot v_t + 1200) / G(z_t \cdot v_t) & \text{if } 1800 < x \cdot v_t \leq z_t \cdot v_t \\ 1 & \text{if } z_t \cdot v_t < x \cdot v_t \end{cases} \quad (2)$$

v_t is the value of `average_output_delay` at the time t that a specific ring is constructed.

z_t is the value of `num_usable_rct_outputs` at the time t that a specific ring is constructed.

2 Counting up unspendable outputs

Let \mathcal{U}_t be the set of outputs at time t that are not spendable due to custom unlock time or the 60 block lock on coinbase outputs. Let u_t be an element of \mathcal{U}_t . We need to find $\dot{\mathbf{u}}_t$, the total the value of the probability mass function at the unspendable points. The probability mass function of the spendable outputs must be scaled up by this total so that the probability mass function sums to one.

Let u_t be in block b_{u_t} . Block b_{u_t} contains outputs with indices y_{0,u_t} to y_{1,u_t} . Therefore, $y_{0,u_t} \leq u_t \leq y_{1,u_t}$. Note that $y_{0,u_t} = u_t = y_{1,u_t}$ is possible because a block can contain a single output, but no fewer than one output because every block must have a coinbase transaction with at least one output. The indices are counted from the first output in the 11th most recent block, starting from index 1.

$$\dot{\mathbf{u}}_t = \sum_{u_t \in \mathcal{U}_t} \frac{G^*(y_{1,u_t} + 1) - G^*(y_{0,u_t})}{y_{1,u_t} + 1 - y_{0,u_t}} \quad (3)$$

3 PMF and CDF in closed form

Let \mathcal{S}_t be the set of spendable outputs at time t . Let s_t be an element of \mathcal{S}_t .

Let s_t be in block b_{s_t} . Block b_{s_t} contains outputs with indices y_{0,s_t} to y_{1,s_t} . Therefore, $y_{0,s_t} \leq s_t \leq y_{1,s_t}$. Note that $y_{0,s_t} = s_t = y_{1,s_t}$ is possible. The probability mass function of the `wallet2` decoy selection algorithm is

$$f(s_t) = \frac{G^*(y_{1,s_t} + 1) - G^*(y_{0,s_t})}{y_{1,s_t} + 1 - y_{0,s_t}} \cdot \frac{1}{1 - \dot{\mathbf{u}}_t} \quad (4)$$

The CDF can be found by computing the cumulative sum of the PMF of spendable outputs:

$$F(s_t) = \sum_{i \in \mathcal{S}_t \text{ s.t. } i \leq s_t} \frac{G^*(y_{1,i} + 1) - G^*(y_{0,i})}{y_{1,i} + 1 - y_{0,i}} \cdot \frac{1}{1 - \dot{\mathbf{u}}_t} \quad (5)$$

4 Example

Below is the PMF for rings in transactions that were constructed and broadcast between the 2,999,999th and 3,000,000th block, which would have been included in the 3,000,000 block in usual circumstances. Just the first 5,000 outputs are shown.

**Probability Mass Function of wallet2 Decoy Selection Algorithm
for Rings in Monero Block 3,000,000**

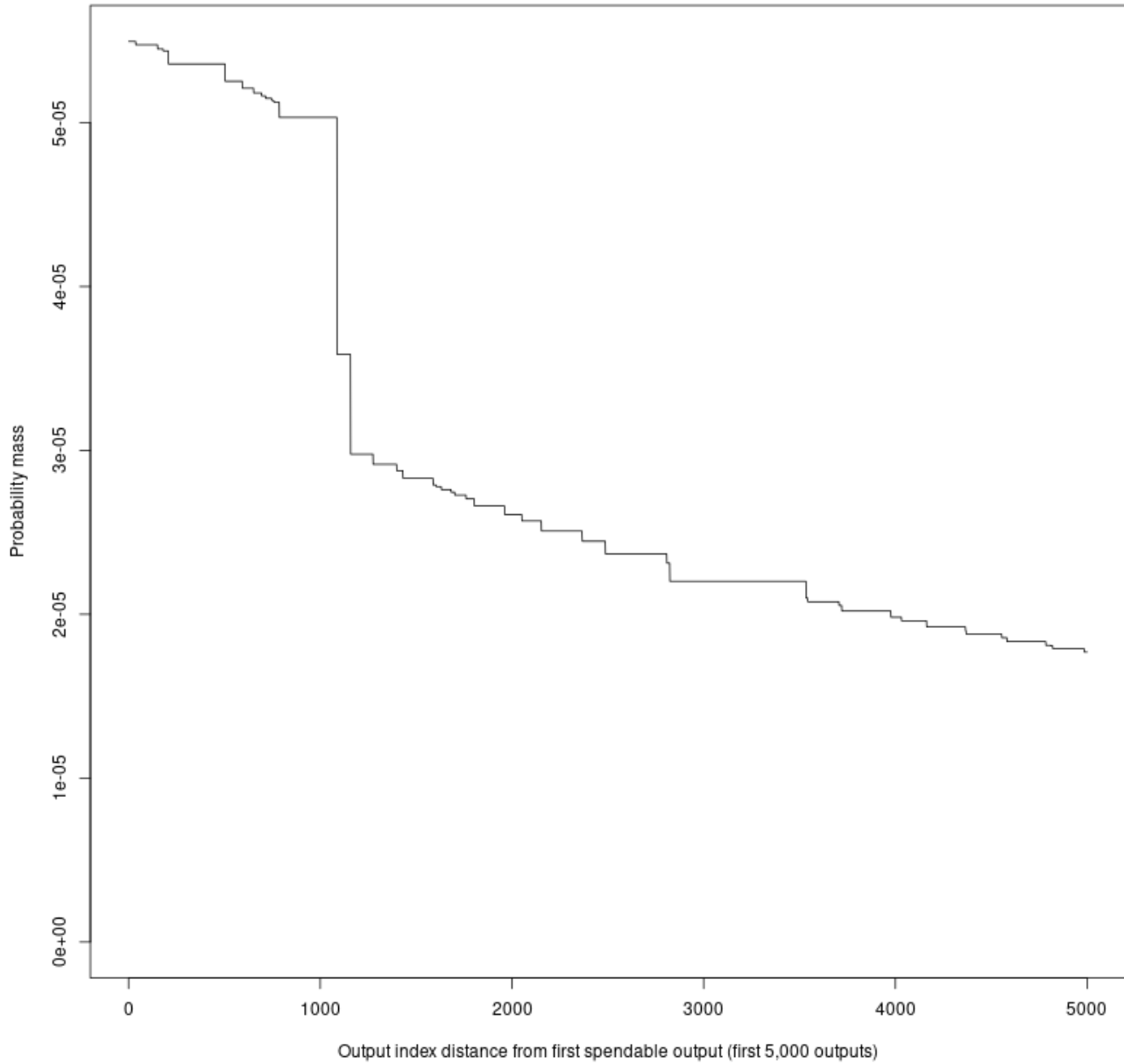


Figure 1: Example PMF