Walker/Knuth alias method

the fastest algorithm in the West for sampling a given probability distribution

Imagine you have a probability space (S,P), where S is a discrete set and P is a probability measure on S, in practice we may enumerate the elements in the set $s_1, s_2, \ldots, s_n, \ldots$

and a probability is defined $p_1, p_2, \ldots, p_n, \ldots$ subject as usual to $p_i \geq 0, \sum p_i = 1$. How can we sample elements from S, say one million of extractions, in a clever way? The simple minded approach like

$r = \text{rand}() \in [0,1);\, if\ r \leq p_1, \text{return } s_1, \text{ elseif } r \leq p_1 + p_2, \text{ return} s_2, \ \ldots \text{end}$

is highly inefficient; it may cost O(N) rand extractions to get just one sample, if N is the number of elements in S. Even if we sort the values of discrete probabilities in descending order this is only going to improve a bit, but just by a prefactor in O(N). In the '70s A.J. Walker [1] introduced a very smart method called the *alias method* or with an intuitive implementation the postmen method. The idea is: imagine we have a post office with L letters to be delivered to N addresses, with L/N=l. The post office head hires a certain amount of people to reach a number N of available postmen. However matters are not so simple: the letters are not evenly distributed, there are l_1 letters to be delivered to the address I_1, and in general l_k to I_k. Moreover the Unions have established a deal according to which each worker *should not reach more than two distinct addresses and the number of envelopes must be the same for all of them!* What a puzzle! However the head of the Post Office is a good friend of a young mathematician who cooks up for him a smart way to solve the problem. One organizes the addresses

[1] J.A.Walker, *An efficient method for generating discrete random variables with general distributions*, ACM Trans. Math. Softw. 1977

sorting them in increasing number of envelopes $l_1 \le l_2 \dots \le l_k \dots \le l_N$. Then we assign l_1 envelopes to the postman P_1 to be taken to the first address; if $l_1 = l = L/N$ then there is an equal distribution of letters among addresses and the problem is already solved. In general $l_1 < l$ and so P_1 can afford to deliver the remaing $l - l_1$ to another address, let's choose I_N. At this point the first worker has a complete list and he can leave. We discard I_1 from the list of addresses and decrease l_N by l_1, sort again the addresses in increasing order of letters and we find ourseves with the same original problem, however the number of addresses is now N-1!. We then apply the same idea recursively and end up with the solution.

Now, it's clear that we can make a ono-to-one correspondence with the problem of extracting L samples from an N-set with probabilities $p_1 \le p_2 \le \dots \le p_n, \dots \le p_N$. The idea is the following: extract an integer *i;* if *i=1* return 1 with probability Np_1, otherwise return A_1 (the *alias)* which is chosen = N. Now decrease p_N by the amount $\frac{1}{N} - p_1$ and sort again the set (2,…,N) according to the new probabilities. We check that 1 will be extracted with probability $\frac{1}{N} N p_1 = p_1$ and N has already been extracted with probability $\frac{1}{N}(1 - Np_1)$ and then its "account" has been decreased accordingly. Now we proceed recursively keeping into account all probabilities $P_j = Np_j$ and the aliases Y_j. We end up with a $2 \times N$ table

$$\begin{pmatrix} P_1 & P_2 & \dots & P_N \\ Y_1 & Y_2 & \dots & Y_N \end{pmatrix}$$

and the algorithm will simply be

1) extract an integer $i \in (1,N)$;
2) extract a real $r = \text{rand}()$;
3) if $r \le P_i$ then return 1 else return Y_i

You see the magic now: however large may N be, a call to randi() and a call to rand() will give you a sample with the correct a priori distribution. This is the best we can help to obtain! A variation on this idea has been devised by M.D. Vose[2] which has its merits, if one wants to go deeper in this theme.

Walker alias method was advertized by D. Knuth in his big work *The Art of Computer Programming.* The method is easily encoded in matlab and you can find the files here under the link **Alias.tar.** There you will find a simple implementation of the method described above; the approach can also be extended to a continuous distribution by a suitable discretization. See ***ikdemo.m*** where the method is applied to several distribution like the Gaussian, Lorentz, Poisson etc.

To use the package one has to define a vector p of dimension N containing the probabilities $p_1, p_2, \ldots, p_n, \ldots$ The vector needs not to be normalized nor sorted, the matlan module *kalias.m* will take care of this. The call

```
>> KAT = kalias(p);
```

returns a structure such that KAT.Y contains the list of aliases and KAT.P the branching probabilities $\{P_i\}$. To extract *Nsample* samples from the distribution we call

```
>> X = Krand(Nsample,KAT,seed);
```

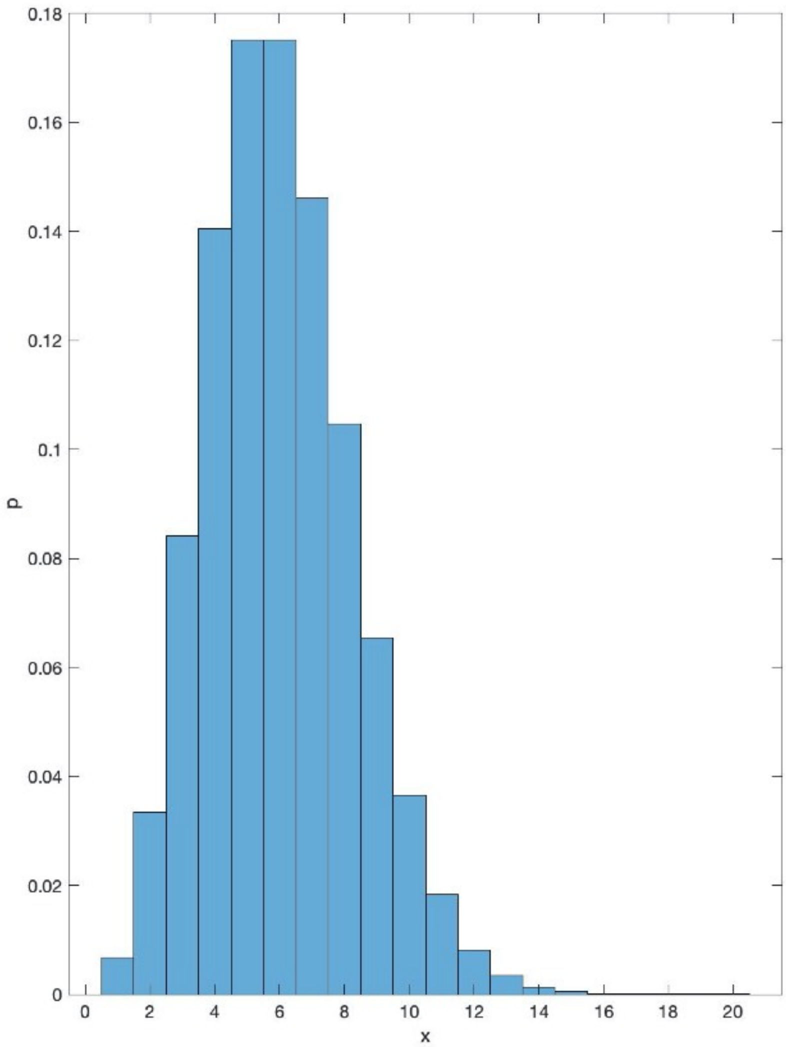
seed is optional, use it if you want to start with a specified seed - see the documentation. Try

```
>> nmax = 50; mu=5;
>> n = 0:nmax;
>> p = mu.^n .* exp(-mu) ./ gamma(n+1);
```

---

[2] A Linear Algorithm For Generating Random Numbers With a Given Distribution, IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. 17, NO. 9, SEPTEMBER 1991

```
>> p = p/sum(p);
```




```
>> KAT = kalias(p);
>> X = Krand(1e6, KAT);
```

In 0.07sec you get the plot