MATLAB Programs for Exact Calculation of Minimum Sample Size for Estimating a Poisson Parameter *

Zhengjia Chen and Xinjia Chen

May 2014

We consider the problem of estimating a Poisson parameter by using both the MLE and range-preserving estimator. We have developed in [3, 5] an exact approach for the determination of the minimum sample size for estimating a Poisson parameter such that the pre-specified levels of relative precision and confidence are guaranteed. The exact computation is made possible by reducing infinitely many evaluations of coverage probability to finitely many evaluations. The theory for supporting such a reduction is that the minimum of coverage probability with respect to the parameter in an interval is attained at a discrete set of finitely many elements. Computational mechanisms have been developed to further reduce the computational complexity. In this document, we provide MATLAB codes for exact calculation of minimum sample size for estimating a Poisson parameter.

1 MATLAB Codes for Computing Sample Size with Maximum Likelihood Estimator

Let X be a Poisson random variable defined in a probability space $(\Omega, \mathscr{F}, \mathbb{P}_\lambda)$ such that the probability mass function is parameterized by its expectation

$$\mathbb{E}[X] = \lambda > 0.$$

The subscript in the notation "$\mathbb{P}_\lambda$" indicates that the probability measure is determined by the Poisson parameter λ. It is a frequent problem to construct an estimator for λ based on n identical

*Dr. Zhengjia Chen is working with Department of Biostatistics and Bioinformatics, Emory University, Atlanta, GA 30322; Email: zchen38@emory.edu. Dr. Xinjia Chen is afflicted with Department of Electrical Engineering and Computer Sciences, Louisiana State University at Baton Rouge, LA 70803; Email: xinjiachen@lsu.edu.

and independent samples of X. A crucial question in the estimation is as follows:

Given the knowledge that a Poisson parameter is contained in interval $[a, b]$, what is the minimum sample size n that guarantees the difference between the Poisson parameter and its estimator be bounded within some prescribed margin of error with a confidence level higher than a prescribed value?

More specifically, let $X_1, \cdots, X_n$ be i.i.d. samples of X. Define sample sum

$$Y_n = \sum_{i=1}^{n} X_i$$

and sample mean

$$\overline{X}_n = \frac{Y_n}{n}.$$

Let $\epsilon \in (0, 1)$ be a margin of relative error. Let $\delta \in (0, 1)$ be a confidence parameter which specifies the confidence level $1 - \delta$. Suppose that the estimator for the Poisson parameter λ is taken as the $\overline{X}_n$. A natural question is as follows.

How large the sample size n should be chosen so that

$$\mathbb{P}_\lambda \left\{ \left| \frac{\overline{X}_n - \lambda}{\lambda} \right| < \epsilon \right\} \geq 1 - \delta \tag{1}$$

for any $\lambda \in [a, b]$?

The most frequently used method for determining sample size for estimating a Poisson parameter is based on normal approximation (see, e.g., [6, 7, 11] and the references therein). Undoubtedly, the normal approximation provides a simple and insightful solution to the relevant sample size problem. However, the normal approximation is an asymptotic method which inevitably introduces unknown error due to the necessary use of a finite number of samples (see, e.g., [9]). In the context of using maximum likelihood estimator $\overline{X}_n$, our MATLAB program for exact computation of the minimum sample size for estimating λ with margin of relative error ϵ and confidence parameter δ is presented as follows.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Function Name: SampleSize.m
% Assume that the Poisson parameter λ is contained in interval [a, b].
% This is the main function for computing the minimum sample size N_min
% with margin of relative error "epsilon" (i.e., ϵ) and confidence parameter "delta" (i.e., δ).
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [N_min] = SampleSize(a, b, delta, epsilon)
```

```
M = 1000000;   % Create a table of ln(n!) with n no greater than M.
Tab = zeros(M+1,1);
for i = 1:nu
      Tab(i+1) = Tab(i) + log(i);
end
[z] = norminv(1 - delta / 2, 0, 1);   % Compute the 100(1 − δ/2)%-quantile
                                        % of the normal distribution,
                                        % which is also called the critical value.
N = ceil( z * z / (a * epsilon * epsilon) );   % Sample size based on normal approximation
enough = 0;   % "enough = 0" means that sample size is not large enough;
               % "enough = 1" means sample size is large enough
while enough == 0
        N = N + 1;
        [enough] = FastCheck(N, a, b, delta, epsilon, Tab);   % Check if the sample size
                                                               % is large enough
end
N_min = N;   % Minimum Sample Size

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Function Name: FastCheck.m
% This function determines if sample size N is large enough
% to ensure that the coverage probability is greater than 1 − δ.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [enough] = FastCheck(N, a, b, delta, epsilon, Tab)
c = 3 / ( N * epsilon * epsilon ) * log(2 / delta);
b = min(b, c);
enough = 1;   % Assume that sample size is enough
ro = floor( N * a * (1 - epsilon) ) + 1;
vro = ceil( N * a * (1 + epsilon) ) - 1;
[U] = ComplementaryCoverage(N, a, ro, vro, Tab);
enough = ( U < delta );   % "enough = 1" means that the sample size is large enough
if enough == 1
      ell = floor(N * a * (1 + epsilon)) + 1;
      q = ell / N / (1 + epsilon);
```

```
        ro = max(0, floor( ell * (1 - epsilon) / (1 + epsilon) ) + 1);
        vro = ell - 1;
        [U] = ComplementaryCoverage(N, q, ro, vro, Tab);
        enough = ( U < delta );
end
while enough == 1 & ell < ceil(N * b) - 1
        ell = ell + 1;
        q = ell / N / (1 + epsilon);
        ro = max(0, floor( ell * (1 - epsilon) / (1 + epsilon) ) + 1);
        vro = ell - 1;
        [U] = ComplementaryCoverage(N, q, ro, vro, Tab);
        enough = ( U < delta );
end
if enough == 1
        ell = floor(N * a ) + 1;
        q = ell / N / (1 - epsilon);
        ro = max(0, ell + 1);
        vro = ceil(ell * (1 + epsilon) / (1 - epsilon)) - 1;
        [U] = ComplementaryCoverage(N, q, ro, vro, Tab);
        enough = ( U < delta );
end
while enough == 1 & ell < ceil(N * b * (1 - epsilon)) - 1
        ell = ell + 1;
        q = ell / N / (1 - epsilon);
        ro = max(0, ell + 1);
        vro = ceil(ell * (1 + epsilon) / (1 - epsilon)) - 1;
        [U] = ComplementaryCoverage(N, q, ro, vro, Tab);
        enough = ( U < delta );
end
if enough == 1
        ro = floor( N * b * (1 - epsilon) ) + 1;
        vro = ceil( N * b * (1 + epsilon) ) - 1;
        [U] = ComplementaryCoverage(N, b, ro, vro, Tab);
        enough = ( U < delta );
```

```
end
if enough == 1
    ro = floor( N * a ) + 1;
    vro = ceil( N * a * (1 + epsilon) / (1 - epsilon) ) - 1;
    [U] = ComplementaryCoverage(N, a / (1 - epsilon), ro, vro, Tab);
    enough = ( U < delta );
end
if enough == 1
    ro = floor( N * b * (1 - epsilon) / (1 + epsilon) ) + 1;
    vro = ceil( N * b ) - 1;
    [U] = ComplementaryCoverage(N, b / (1 + epsilon), ro, vro, Tab);
    enough = ( U < delta );
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Function Name: ComplementaryCoverage.m
% This function computes tight upper bound U
% for the complementary coverage probability for λ = q.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [U] = ComplementaryCoverage(N, q, ro, vro, Tab)
U = 0;
if ro > 0
    [U] = BoundPoissonSum(N, q, 0, ro - 1, Tab);
end
[V] = BoundPoissonSum(N, q, vro + 1, inf, Tab);
U = U + V;    % upper bound of complementary coverage probability

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Function Name: BoundPoissonSum.m
% This function computes tight upper bound for the Poisson sum
% ∑_{k=k1}^{k2} e^{-Nλ}(Nλ)^k / k! by using the Chernoff bound [2] and
% the truncation technique proposed in [4].
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [Pru] = BoundPoissonSum(N, lm, k1, k2, Tab)
```

```
eta = 0.00000001;    % tolerance of truncation error
k1 = max(0, k1);
if lm == 0
    Pru = 0;
    if k1 == 0 & k1 <= k2
        Pru = 1;
    end
else
    [rl, ru] = TruncatePoissonSum(N, lm, eta);
    m1 = max(k1, rl);
    m2 = min(k2, ru);
    [Pr] = PoissonSum(N, lm, m1, m2, Tab);
    Pru = min(1, Pr + eta);    % upper bound for the Poisson sum
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Function Name: PoissonSum.m
% This function computes the Poisson sum sum_{m=m1}^{m2} e^{-n*lambda}(n*lambda)^m / m!.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [Pr] = PoissonSum(n, lm, m1, m2, Tab)
lm = n * lm;
if m1 > m2
    Pr = 0;
else
    if lm == 0
        Pr = ( m1 == 0 );
    else
        Pr = 0;
        for m = m1: m2
            Pr = Pr + exp( - lm - Tab(m + 1) + m * log(lm) );
        end
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
% Function Name: TruncatePoissonSum.m
% This function reduces the number of terms of Poisson sum
% by the truncation technique proposed in [4].
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [rl, ru] = TruncatePoissonSum(n, lm, eta)
lm = n * lm;
eta = log(eta / 2);
% First compute the upper limit
r = lm;
stop = 0;
while stop == 0
    r = r * 2;
    if r * ( 1 + log(lm / r) ) - lm <= eta;
        stop = 1;
    end
end
rmax = r;
rmin = lm;
while rmax - rmin > 1
    r = (rmin + rmax) / 2;
    if r * ( 1 + log(lm / r) ) - lm <= eta;
        rmax = r;
    else
        rmin = r;
    end
end
ru = ceil(rmax);
% Now compute the lower limit
if - lm > eta
    rl = -1;
else
    rmin = 0;
    rmax = lm;
    while rmax - rmin > 1
```

```
            r = (rmin + rmax) / 2;
            if r * ( 1 + log(lm / r) ) - lm <= eta;
                rmin = r;
            else
                rmax = r;
            end
        end
        rl = floor(rmin);
    end
```

2 MATLAB Codes for Computing Sample Size with Range-Preserving Estimator

When the Poisson parameter λ is known to be bounded in the interval $[a, b]$, it may also be appropriate to use the range-preserving estimator (see, e.g. [1, 8, 10, 12, 13, 14] and the references therein). Specifically, define

$$\widehat{\boldsymbol{\lambda}}_n = \begin{cases} \overline{X}_n & \text{for } \overline{X}_n \in [a, b], \\ a & \text{for } \overline{X}_n < a, \\ b & \text{for } \overline{X}_n > b. \end{cases}$$

Let $\epsilon \in (0, 1)$ be the margin of relative error and $\delta \in (0, 1)$ be the confidence parameter. It is desirable to determine the minimum sample size n so that

$$\mathbb{P}_\lambda \left\{ \left| \frac{\widehat{\boldsymbol{\lambda}}_n - \lambda}{\lambda} \right| < \epsilon \right\} \geq 1 - \delta \tag{2}$$

for any $\lambda \in [a, b]$. Clearly, to obtain the exact value of the minimum sample size, an essential task is to determine whether (2) is guaranteed for any $\lambda \in [a, b]$ for a given value of sample size n.

In the context of using the range-preserving estimator, our MATLAB program for computing the minimum sample size for estimating λ with margin of relative error ϵ and confidence parameter δ is the same as that in the context of using the maximum likelihood estimator except that the function FastCheck.m is revised as follows.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Function Name: FastCheck.m
% This function determines if sample size N is large enough
% to ensure that the coverage probability is greater than 1 − δ.
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [enough] = FastCheck(N, a, b, delta, epsilon, Tab)
c = 3 / ( N * epsilon * epsilon ) * log(2 / delta);
b = min(b, c);
enough = 1;    % Assume that sample size is enough
ro = floor( N * a * (1 - epsilon) ) + 1;
vro = ceil( N * a * (1 + epsilon) ) - 1;
[U] = ComplementaryCoverage(N, a, ro, vro, Tab);
enough = ( U < delta );    % "enough = 1" means that the sample size is large enough
if enough == 1
    ell = floor(N * a * (1 + epsilon)) + 1;
    q = ell / N / (1 + epsilon);
    ro = max(0, floor( ell * (1 - epsilon) / (1 + epsilon) ) + 1);
    vro = ell - 1;
    [U] = ComplementaryCoverage(N, q, ro, vro, Tab);
    enough = ( U < delta );
end
while enough == 1 & ell < ceil(N * b * (1 + epsilon)) - 1
    ell = ell + 1;
    q = ell / N / (1 + epsilon);
    ro = max(0, floor( ell * (1 - epsilon) / (1 + epsilon) ) + 1);
    vro = ell - 1;
    [U] = ComplementaryCoverage(N, q, ro, vro, Tab);
    enough = ( U < delta );
end
if enough == 1
    ell = floor(N * a * (1 - epsilon)) + 1;
    q = ell / N / (1 - epsilon);
    ro = max(0, ell + 1);
    vro = ceil(ell * (1 + epsilon) / (1 - epsilon)) - 1;
    [U] = ComplementaryCoverage(N, q, ro, vro, Tab);
    enough = ( U < delta );
end
while enough == 1 & ell < ceil(N * b * (1 - epsilon)) - 1
```

```
        ell = ell + 1;
        q = ell / N / (1 - epsilon);
        ro = max(0, ell + 1);
        vro = ceil(ell * (1 + epsilon) / (1 - epsilon)) - 1;
        [U] = ComplementaryCoverage(N, q, ro, vro, Tab);
        enough = ( U < delta );
    end
```

References

[1] Blyth, C. R. (1993), "Restrict estimates to the possible values?" *The American Statistician*, 47, 73-75.

[2] Chernoff, H. (1952), "A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations," *Annals of Mathematical Statistics*, 23, 493-507.

[3] Z. Chen and X. Chen, "Exact Calculation of Minimum Sample Size for Estimating a Poisson Parameter," submitted for publication.

[4] X. Chen, "A truncation approach for fast computation of distribution functions," arXiv:0802.3455 [math.ST], February 2008.

[5] X. Chen, "Exact computation of minimum sample size for estimation of Poisson parameters," arXiv:0707.2116v1 [math.ST], July 2007.

[6] Chow, S. C., Shao, J., and Wang, H. (2008), *Sample Size Calculations in Clinical Trials*, 2nd edition, Chapman & Hall.

[7] Desu, M. M., and Raghavarao, D. (1990), *Sample Size Methodology*, Academic Press.

[8] Gamrot, W. (2013), "On exact computation of minimum sample size for restricted estimation of a binomial parameter," *Journal of Statistical Planning and Inference*, 143, 852-866.

[9] Hampel, F. (1998), "Is statistics too difficult?" *The Canadian Journal of Statistics*, 26, 497-513.

[10] Hoeffding, W. (1983), "Unbiased range-preserving estimators," In: Bickel, P. J., Doksum, K., Hodges, J. L., Jr.(Eds.), *A Festschrift for Erich L. Lehmann*. Wadsworth, Belmont, pp. 249-260.

[11] Johnson, N. L., Kemp, W. A., and Kotz, S. (1993), *Univariate Discrete Distributions*, 3rd edition, Wiley.

[12] Marchand, E., and Strawderman, W. E. (2004), "Estimation in restricted parameter spaces: a review," In: *A Festschrift for Herman Rubin*, Institute of Mathematical Statistics, Lecture Notes-Monograph Series, 45, 21-44.

[13] Sibuya, M. (1963), "Randomized unbiased estimation of restricted parameters," *Annals of the Institute of Statistical Mathematics*, 15, 61-66.

[14] Sahai, H., and Khurshid, A. (1993), "Confidence intervals for the mean of a Poisson distribution: A review," *Biometric Journal*, 7, 857-867.