J Glob Optim (2006) 36:319–338
DOI 10.1007/s10898-006-9011-6

ORIGINAL ARTICLE

Using global optimization to estimate population class sizes

Betsy S. Greenberg · Leon S. Lasdon

Received: 4 July 2003 / Accepted: 21 February 2006 / Published online: 27 June 2006

Abstract In this paper we formulate a nonlinear optimization model to estimate population class sizes based on sample information. The model is nonconvex and has several local minima corresponding to different populations that could have been the source of the sample data. We show that many if not all local solutions can be found using a new global optimization algorithm called OptQuest/NLP (OQNLP). This can be used to estimate the number of individuals in a population with unique or rarely occurring characteristics, which is useful for assessing disclosure risk. It can also be used to estimate the number of classes in a population, a problem with applications in a variety of disciplines.

Keywords Global optimization · Confidentiality · Disclosure risk · Number of species · Nonlinear programming · Microdata · Statistical estimation · Computational experiments

1 Introduction

Government agencies collect and release data related to a variety of topics including income, unemployment, crime, and healthcare. The data are used by policy makers and to meet the needs of researchers. Data collection in the private sector has also soared. For example, corporations now collect information on shopping patterns of hundreds of millions of consumers. As more information is collected, the public has become increasingly concerned about confidentiality.

Before a data set is released, obvious identifying information such as name, address, telephone number, etc. are deleted. However, this may not be sufficient to prevent disclosure. Willenborg and de Waal (2001) illustrate the problem with an example of an academic economist studying data from a government survey of income and expenditure. If the economist happens to discover data about a female dentist living in a specified area and he also happens

B.S. Greenberg (✉) · L.S. Lasdon
IROM Department, McCombs School of Business
University of Texas at Austin
Austin, TX 78712, USA
e-mail: betsy.greenberg@mccombs.utexas.edu

to know there is only one such dentist in the area, then he could misuse sensitive information. This type of disclosure occurs when an individual has a set of characteristic values that are unique both in the data set and in the population. If there were actually two or three female dentists in the area, but the economist believes there is only one, it is still possible to misuse sensitive information. Therefore, data coming from rarely occurring sets of values has the potential for misuse.

To protect confidentiality, it is necessary to distort or summarize the data before it is released. Disclosure control techniques include recoding (Hurkens and Tiourine 1998), cell suppression (de Waal and Willenborg, 1998), data swapping (Dalenius and Reiss 1982), adding noise (Kim 1986), and rounding (Dalenius 1981). These techniques prevent individuals from being identified, but they also involve some loss of information. In order to balance the conflicting goals of protecting confidentiality (avoiding disclosure) and providing high quality information, it is important to accurately assess disclosure risk.

In this paper, we formulate an optimization model to estimate population class size distributions based on sample information. Usually, disclosure risk is assessed by estimating the number of individuals in a population with unique characteristics. This ignores the disclosure risk for individuals with rarely occurring, but not necessarily unique characteristics. Since we estimate all class sizes, the results can be used to better assess disclosure risk.

The results from our optimization procedure can also be used to estimate the number of classes in a finite population. This is a problem that has been studied for over 60 years by researchers from a variety of disciplines. A review by Bunge and Fitzpatrick (1993) on the problem lists over 125 references. Applications range from biologists and ecologists interested in estimating the number of species in plant or animal populations to linguists interested in estimating the size of an author's vocabulary (Efron and Thisted 1976). Our results can be used to estimate the number of unduplicated signatures on a petition (Smith-Cayama and Thomas 1999), the number of distinct names on a combination of administrative lists (Madigan and York 1997), or the number of distinct values of each attribute in a relational database (Haas et al. 1995).

Various models have been proposed in the literature for the purpose of estimating the number of population uniques from a sample of data. A Poisson-Gamma (Bethlehem et al. 1980) model, Poisson-lognormal (Skinner and Holmes 1993) model, negative binomial (Chen and Keller-McNulty 1998) model, Dirichlet-multinomial (Takemura 1999) model, and a generalization of the Dirichlet multinomial model based on Pitman's sampling formula (Hoshino 2001) have all been proposed. Greenberg and Zayatz (1992) proposed a procedure that is not dependent on a model for the population of class sizes. Unfortunately, none of these approaches have been totally successful. Each method may work well for one population, but work poorly for another with different characteristics. The problem of estimating the number of species in the population also does not have a solution that works well for all populations.

The problem of estimating population class sizes is difficult because, especially as the sampling fraction decreases, samples from two or more very different populations can be nearly identical. The optimization model we develop is nonconvex and has several local minima corresponding to different populations that could have been the source of the sample data. We show that many if not all local solutions can be found using a new global optimization algorithm called OptQuest NLP (OQNLP). Section 2 of this paper describes Greenberg's (2003) recursive algorithm and its performance. The non-linear program is developed in Sect. 3. Section 4 describes the OQNLP algorithm and the GAMS modeling language used to pose our test problems. Section 5 provides computational results, with conclusions in Section 6.

2 Recursive algorithm

Let μ_i and p_i be respectively the number and proportion of classes of size i in the population. That is,

$$p_i = \mu_i \Big/ \sum_{all\ i} \mu_i. \tag{1}$$

Let m_i be the number of classes of size i in the sample. Greenberg's (2003) algorithm finds a solution to (1) and the following two sets of equations:

$$P(i_p|j_s) = \frac{p_i P(j_s|i_p)}{\sum_{j=1}^{i} p_i P(j_s|i_p)}, \tag{2}$$

and

$$\mu_i = \frac{\sum_{j=1}^{i} m_j P(i_p|j_s)}{1 - P(0_s|i_p)}, \tag{3}$$

where $P(j_s|i_p)$ is a hypergeometric probability:

$$P(j_s|i_p) = \frac{\binom{i_p}{j_s}\binom{N_p - i_p}{N_s - j_s}}{\binom{N_p}{N_s}}, \tag{4}$$

where N_p is the size the of population and N_s is the size of the sample. $P(i_p|j_s)$ is the probability that a class of size j in the sample appears as a class of size i in the population and can be calculated using Bayes' Rule in (2). Equation (3) is a method of moments estimate for μ_i. $P(j_s|i_p)$, μ_i, and p_i are solved for recursively as follows.

Step 1: Estimate p_i using the sample data, $\hat{p}_i = m_i \Big/ \sum_{all\ i} m_i$.
Step 2: Solve for $P(i_p|j_s)$ using (2).
Step 3: Solve for μ_i using (3).
Step 4: Solve for p_i using (1).
Return to Step 2 and continue until the procedure converges.

The upper limit M on the sums in (1), (2), and (3) is selected so that

$$\sum_{i=1}^{M} i\mu_i = N_p, \tag{5}$$

is satisfied as closely as possible. This is done by initially letting M be equal to the largest class size in the sample. M can be increased until (5) is satisfied or nearly satisfied.

Greenberg (2003) demonstrated the recursive algorithm for some challenging examples suggested by Greenberg and Zayetz (1992). One artificial population (P_2) considered had 1000 uniques and 900 groups of 10($\mu_1 = 1000, \mu_2 = \cdots = \mu_9 = 0, \mu_{10} = 900$). When sampling rates are moderate to high, the recursive algorithm performs very well. Results for 50, 30, and 20% sampling rates are in Tables 1–3 in the line labeled "Recursive". However,

Table 1 Example with 50% sample from P_2

	obj	Class sizes									
		1	2	3	4	5	6	7	8	9	10
Population		1000	0	0	0	0	0	0	0	0	900
Sample		513	41	100	182	248	167	89	48	12	2
Recursive		1006	0	0	0	0	0	0	0	270	371
L_1,1	0.52	1008	0	0	0	0	0	0	0	0	899
error		0.52	0	0	0	0	0	0	0	0	0

Table 2 Example with 30% sample from P_2

	obj	Class sizes									
		1	2	3	4	5	6	7	8	9	10
Population		1000	0	0	0	0	0	0	0	0	900
Sample		424	208	243	191	79	31	10	2	0	0
Recursive		1044	0	0	0	0	0	0	0	53	848
L_1,1	1.1E−4	1044	0	0	0	0	0	0	0	52	849
L_1,2	2.5E−1	0	0	0	0	0	0	110	1154	0	0

Table 3 Example with 20% sample from P_2

	obj	Class sizes										Norms	
		1	2	3	4	5	6	7	8	9	10	standard	shift
Population		1000	0	0	0	0	0	0	0	0	900		
Sample		436	288	185	75	19	4	2	0	0	0		
Recursive		837	0	0	0	0	0	0	0	374	580		
L_1,1	3.26E−4	778	0	0	0	0	102	0	0	153	724	0.25	0.31
L_1,2	9.7E−3	0	594	0	0	0	0	0	0	135	760	0.88	0.73
L_1,3	1.3E−2	0	0	0	0	0	0	401	0	799	0	1.2	1.28

when sampling rates are lower, Greenberg showed that the recursive algorithm may converge to a population that could have generated the sample, but not necessarily the one that did. Table 4 shows that for a 10% sample from P_2 ($m_1 = 449$, $m_2 = 174$, $m_3 = 52$, $m_4 = 10$, $m_5 = 1$) the recursive algorithm converged to a solution with no uniques ($\mu_8 = 224$, $\mu_9 = 913$). This is not surprising, since a 10% sample from this population (called P_3, and shown in Table 5), ($m_1 = 440$, $m_2 = 191$, $m_3 = 48$, $m_4 = 8$, $m_5 = 1$) can look very much like the 10% sample from P_2.

In situations where more than one population can generate the same sample, it will be very difficult (if not impossible) to find a method guaranteed to work well unless additional information is used. Greenberg (2003) showed that when prior information is available, a starting point different from the one based on the sample could be used, resulting in a much better solution. That is, the recursive algorithm converged to different solutions depending on the starting point. In the absence of reliable prior information, it should be helpful to arbitrarily choose a variety of starting points to see whether alternate solutions exist. In practice, although we can't determine which solution is correct, it is useful in assessing disclosure risk to know that the data *may* have come from two (or more) different populations.

Table 4 Example with 10% sample from P_2

	obj	Class Sizes										Norms	
		1	2	3	4	5	6	7	8	9	10	standard	shift
Population		1000	0	0	0	0	0	0	0	0	900		
Sample		449	174	52	10	1	0	0	0	0	0		
Recursive		0	0	0	0	0	0	0	224	913	0		
Recursive2		935	0	0	0	0	0	0	0	111	807		
$L_{1,1}$	3.0E−4	952	0	0	0	0	0	0	0	82	831	0.09	0.06
$L_{1,2}$	1.51E−2	0	0	0	0	0	0	538	0	0	624	0.87	1.07
$L_{1,3}$	1.66E−2	485	310	0	0	0	0	0	0	0	890	0.45	0.35
$L_{1,4}$	1.83E−2	0	0	0	0	0	0	0	243	895	0	1.21	1.06

Table 5 Example with 10% sample from P_1

	obj	Class Sizes								
		1	2	3	4	5	6	7	8	9
Population		0	0	0	0	0	0	0	224	913
Sample		440	191	48	8	1	0	0	0	0
Recursive		0	0	0	0	0	0	0	232	906
$L_{1,1}$	2.5E−06	0	0	0	10	0	0	97	21	1014
$L_{1,2}$	4.7E−06	0	0	0	8	0	0	113	0	1021
$L_{1,3}$	6.0E−06	0	0	0	25	0	0	0	153	965
$L_{1,4}$	8.6E−06	0	0	0	21	0	0	27	116	979
$L_{1,5}$	1.4E−05	0	0	0	0	37	0	0	123	982
$L_{1,15}$	4.7E−06	0	0	0	8	0	0	113	0	1021

3 Formulation as a nonlinear optimization problem

3.1 Basic formulation

For a fixed value of M, the recursive procedure described above is the well known Gauss–Seidel algorithm (iteration by substitution) applied to the problem of solving the nonlinear equations (1)–(3), with the added step of increasing M if the calculated population size is sufficiently smaller than the actual size. This algorithm has rather weak convergence properties: convergence is guaranteed if the solution exists and is unique, the initial guess is in some neighborhood of the solution, and the Jacobian matrix of the nonlinear functions is diagonally dominant (White and Sangiovanni-Vincentelli 1987.). It may also converge to different final points from different initial points. Since, the ability to identify many possible solutions is desirable when the sampling rate is low, we have developed an alternative solution procedure which uses an optimization formulation, and automatically finds multiple local solutions when they exist. We allow errors in the estimation equations (3) and minimize some norm of these errors. Inequality constraints are added, insuring that classes of size j in the sample arise from classes of that or larger size in the population. A multi-start global optimization algorithm is applied to this problem, with its options set to locate many local solutions.

To eliminate p_i from our formulation, we substitute (1) into (2), yielding

$$P(i_p|j_s) = \mu_i P(j_s|i_p) / \sum_{k \geq j} \mu_k P(j_s|k_p), \text{ for all } (i, j)$$
$$\text{with } i \leq M,\ j \leq M,\ j \leq i \tag{6}$$

The relaxed estimation equations (3) are:

$$\mu_i[1 - P(0_s|i_p)] - \sum_{j=1}^{i} m_j P\left(i_p|j_s\right) = pdev_i - ndev_i, \quad i = 1, \ldots, M \tag{7}$$

where $pdev_i$ and $ndev_i$ are nonnegative "deviation" variables representing the positive and negative errors in the ith equation.

To reflect the fact that classes of size j in the sample, must arise from classes of size j or higher in the population, we impose the constraints

$$\sum_{i \geq j} \mu_i \geq m_j, \quad j = 1, \ldots, M \tag{8}$$

In addition, all variables μ_i, $P(i_p|j_s)$, $pdev_i, ndev_i$ must be nonnegative. The objective is to minimize some norm of the residuals in equation (7). If the L_1 norm (the sum of absolute residuals) is used, the objective is:

$$\text{minimize} \qquad abserr = \sum_{i=1}^{M} (pdev_i + ndev_i) \tag{9}$$

Alternatively, if the L_2 norm is used, the objective would be

$$\text{minimize} \qquad sqerr = \sum_{i=1}^{M} (pdev_i + ndev_i)^2 \tag{9.1}$$

If the $L_{\max}$ norm is used, the objective would be

$$\text{minimize} \qquad maxerr = \max_i (pdev_i, ndev_i) \tag{9.2}$$

The formulation with the L_2 and L_{max} objective will be discussed in Subsection 3.3 below, where a formulation which avoids the nondifferentiable *max* function in (9.2) is provided.

Any solution to this problem must have at least one of each pair of deviation variables equal to zero, otherwise *abserr*, *sqerr*, or *maxerr* could be reduced without violating any constraints by subtracting the smallest of each pair from both variables. Hence, in any optimal solution, the sum of positive and negative deviation variables equals the absolute residual of the equation in which those variables appear.

The resulting model minimizing (9) subject to (5)–(8), that is

$$\begin{aligned}
\text{minimize} \qquad & abserr = \sum_{i=1}^{M} (pdev_i + ndev_i) \\
& \text{subject to} \sum_{i=1}^{M} i\mu_i = N_p, \\
& P(i_p|j_s) = \mu_i P(j_s|i_p) / \sum_{k \geq j} \mu_k P(j_s|k_p), \\
& \qquad\qquad \text{for all } (i, j) \text{ with } i \leq M,\ j \leq M,\ j \leq i \\
& \mu_i[1 - P(0_s|i_p)] - \sum_{j=1}^{i} m_j P\left(i_p|j_s\right) \\
& \quad = pdev_i - ndev_i, \quad i = 1, \ldots, M \\
& \sum_{i \geq j} \mu_i \geq m_j, \quad j = 1, \ldots, M
\end{aligned}$$

has $M(M-1)/2 + 3M$ variables with non-negativity constraints and $M(M-1)/2 + 2M + 1$ additional constraints. Eliminating $P\left(i_p|j_s\right)$ can reduce the model size. To do this, define $\mathrm{E}m_j$, the expected number of classes of size j in the sample:

$$\mathrm{E}m_j = \sum_{i=j}^{M} \mu_i P(j_s|i_p) \quad j = 1, \ldots, M \tag{10}$$

Substitute (10) into (6) to obtain

$$P(i_p|j_s) = \mu_i P(j_s|i_p)/\mathrm{E}m_j, \quad \text{for all } (i, j) \text{ with } j \le i \le M$$

which we substitute into (7) to eliminate $P\left(i_p|j_s\right)$. We obtain

$$\mu_i[1 - P(0_s|i_p)] - \mu_i \sum_{j=1}^{i} m_j \frac{P\left(j_s|i_p\right)}{\mathrm{E}m_j} = pdev_i - ndev_i, \quad i = 1, \ldots, M$$

which we simplify by noting that $\sum_{j=0}^{i} P\left(j_s|i_p\right) = 1$. Finally, we obtain

$$\mu_i \sum_{j=1}^{i} P\left(j_s|i_p\right)\left(1 - \frac{m_j}{\mathrm{E}m_j}\right) = pdev_i - ndev_i, \quad i = 1, \ldots, M \tag{11}$$

It is necessary to impose small positive lower bounds on $\mathrm{E}m_j$ because it appears in the denominator in (11):

$$\mathrm{E}m_j \ge eps \quad j = 1, \ldots, M \tag{12}$$

where $eps = 1.\mathrm{E}-10$ has sufficed for our experiments with maximum class sizes up to 21. These are needed because the values of $\mathrm{E}m_j$ defined by (10) approach zero as j increases, and are smaller than $1.\mathrm{E}-10$ for $M > 20$.

The final formulation minimizes (9), subject to (5), (8), (10), (11) and (12), and is given below:

$$\text{minimize} \quad abserr = \sum_{i=1}^{M} (pdev_i + ndev_i) \tag{9}$$

subject to

$$\sum_{i=1}^{M} i\mu_i = N_p, \tag{5}$$

$$\sum_{i \ge j} \mu_i \ge m_j, \quad j = 1, \ldots, M \tag{8}$$

$$\mathrm{E}m_j = \sum_{i=j}^{M} \mu_i P(j_s|i_p) \quad j = 1, \ldots, M \tag{10}$$

$$\mu_i \sum_{j=1}^{i} P\left(j_s|i_p\right)\left(1 - \frac{m_j}{\mathrm{E}m_j}\right) = pdev_i - ndev_i, \quad i = 1, \ldots, M \tag{11}$$

$$\mathrm{E}m_j \ge eps \quad j = 1, \ldots, M \tag{12}$$

This model has $4M$ variables with non-negativity constraints and $4M + 1$ additional constraints, and is used in all our computational experiments.

3.2 Accounting for an unknown largest class size

Let $cmax$ denote the true largest population class size, and M be the upper limit of all sums over i and j in (5), (8), (10)–(12). Unfortunately, $cmax$ is often unknown. We know that $cmax$ is at least as large as the size of the largest class in the sample, so this can be used as an initial guess for M. Because the deviation variables allow errors of arbitrary size in (11), the problem may have feasible solutions even if $M < cmax$. Our computational experience has shown that the optimal value of the objectives (9) is often much larger if $M < cmax$ than for $M \geq cmax$, as discussed in the next section, signaling that M must be increased. Once a small error norm has been achieved, potential solutions can be found with that value of M or larger values.

3.3 Minimizing the sum of squared errors and the maximum absolute error

A more compact formulation with only linear constraints arises if the L_2 norm of the errors in (9.1) is minimized, rather than the L_1 norm. The nonlinear constraints (11) and the variables $pdev_i$ and $ndev_i$ can be eliminated, because the sum of squares of errors may be written

$$\sum_{i=1}^{M} (pdev_i - ndev_i)^2 = \sum_{i=1}^{M} \mu_i^2 \left(\sum_{j=1}^{i} p(j_s|i_p)(1 - m_j/Em_j) \right)^2 \tag{13}$$

This objective is minimized subject to the nonnegativities on the μ_i, the single linear population constraint (5), the linear constraints defining the variables Em_j (10), the lower bounds on Em_j (12), and the linear inequalities (8). This linearly constrained problem will be more easily solved by gradient-based NLP solvers than the nonlinearly constrained L_1 norm formulation. However the objective (13) is nonconvex, and both L_1 and L_2 formulations have about the same number of local minima in our test problems.

A formulation that minimizes the maximum absolute error in (11) is also useful, because its solutions have the most uniform distribution of errors, and tend to have smoother class size distributions μ. We define a new variable, z, whose optimal value is the max absolute error, drop the variables $pdev_i$ and $ndev_i$, and replace (11) by

$$-z \leq \mu_i \sum_{j=1}^{i} P(j_s|i_p)[1 - m_j/Em_j] \leq z, \quad i = 1, \ldots, M \tag{14}$$

The objective is now to simply

$$\text{Minimize} \quad z \tag{15}$$

subject to (14) and the linear constraints (5), (8), (10), and (12).

3.4 Choppy estimates and smoothness constraints

Often, optimal solutions of the above NLP estimation problem minimizing the L_1 norm have many components of the errors (where $error_i = pdev_i - ndev_i$) in the estimator equations (11) equal to zero, a common characteristic of least absolute value approximations. This causes many components of the estimate vector $\hat{\mu}_1, \ldots, \hat{\mu}_M$ to be zero. Examination of (11)

shows that the error in the *i*th estimator equation is the product of μ_i and the weighted sum of deviations between the expected sample for the population μ and the actual sample. Hence if the *i*th error is nonzero, μ_i must be nonzero. If the *i*th error is zero, μ_i can be nonzero if the second factor in the product is zero. This is illustrated in Table 1, which lists the error term for each i in the bottom row. For $i = 10$, $\mu_{10} = 899$ even though the error is zero. However it is rare for the second term to vanish, so zero errors usually imply zero values for μ_i, as occurs in the second to ninth best solutions to the problem in Table 1.

This "choppiness" leads to poor estimates when the true class size distribution is smooth. This can be remedied by imposing smoothness constraints on the μ_i. Let x be the class size, temporarily allowed to have any real value between 0 and M, and let $\mu(x)$ be the number of classes of size x in the population. We impose smoothness by constraining $\mu(x)$ to be a cubic spline over the interval $[0, M]$, with a knot at the midpoint of this interval, denoted by $\overline{x} = M/2$. Hence

$$\mu(x) = \mu_1(x), \quad 0 \le x \le \overline{x} \tag{16}$$

$$\mu(x) = \mu_2(x), \quad \overline{x} \le x \le M \tag{17}$$

where

$$\mu_k(x) = a_k + b_k x + c_k x^2 + d_k x^3, k = 1, 2 \tag{18}$$

The coefficients of each cubic are constrained so that the value and first 2 derivatives of the 2 polynomials are equal at the knot, $\overline{x}$, which leads to three linear constraints on the coefficients:

$$\mu_1(\overline{x}) = \mu_2(\overline{x}), \mu_1'(\overline{x}) = \mu_2'(\overline{x}), \mu_1''(\overline{x}) = \mu_2''(\overline{x}), \tag{19}$$

These are easily handled by the NLP solvers we have used. The variables μ_i are now denoted by $\mu(i)$, the polynomial coefficients ((a_k, b_k, c_k, d_k) are additional variables, and we add the constraints (18)–(19), with x evaluated at all integral values, i, between 0 and M.

3.5 Error norms

In evaluating the quality of a particular solution, the standard norms of the difference between estimated and true class size distributions can be misleading. To illustrate this, consider the actual and two estimated size distributions shown in Table 6. The actual and the estimated solutions each have 70 elements. The column labeled 'standard' contains the standard norm, i.e. $\sum_i |\mu_i - \hat{\mu}_i|$ divided by $\sum_i \mu_i$, where μ_i is the actual class size and $\hat{\mu}_i$ is the estimated class size. The standard norm is 2 for both estimates, but estimate 1 is "closer" to the true distribution, because the elements in its incorrect classes need be shifted fewer positions to make it correct than those of estimate 2. The column labeled 'Shift norm' is a norm we propose that takes positions shifted into account. We define a penalty cost $c_{ij} = |i - j|/i$

Table 6 Comparison of standard and shift error norms

Class size	1	2	3	4	5	6	Norms	
							standard	shift
Actual	10	0	0	0	0	10		
Estimate 1	0	10	0	0	10	0	2	1.75
Estimate 2	0	0	10	10	0	0	2	2.33

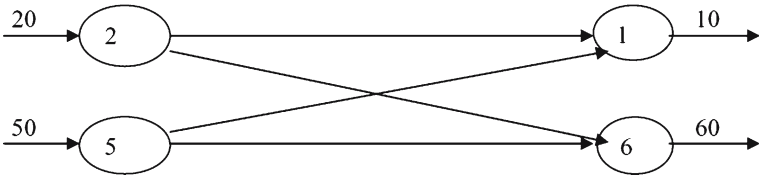

Fig. 1 Transportation network for shift norm

which takes into account the distance of the shift as well as imposing a higher penalty for shifts from smaller classes. The shift norm is defined as the optimal objective value for the following transportation problem divided by $\sum_i \mu_i$.

$$\begin{aligned} &\min \sum_{i,j} c_{ij} x_{ij} \\ &\text{Subject to} \sum_j x_{ij} = i\hat{\mu}_i \qquad \text{(supply constraint)} \\ &\qquad \sum_i x_{ij} = j\mu_j. \qquad \text{(demand constraint)} \end{aligned}$$

For estimate 1, the shift norm is illustrated with the transportation network shown in Fig. 1. The coefficients $\hat{\mu}_2 = 10$, $\hat{\mu}_5 = 10$, $\mu_1 = 10$, and $\mu_6 = 10$ result in supplies of 20 at node 2 and 50 at node 5 and demands of 10 at node 1 and 60 and node 6. The penalty costs are $c_{21} = \frac{1}{2}$, $c_{26} = \frac{4}{2}$, $c_{51} = \frac{4}{5}$, and $c_{56} = \frac{1}{5}$.

The optimal solution values are $x_{2,1} = 10$, $x_{2,6} = 10$, $x_{5,6} = 50$. The optimal objective value is 35, so the shift error norm is equal to 1.75. It is easy to verify that for estimate $2\left(\hat{\mu}_3 = 10, \hat{\mu}_4 = 10\right)$, the optimal solution values are $x_{31} = 10$, $x_{36} = 20$, $x_{46} = 40$, the optimal objective value is 46.67, so the shift error norm is equal to 2.33.

4 Coding and solving the nonlinear optimization model

The only nonlinearities in the L_1 model arise from the quotient terms μ_i / Em_j in the estimator equations (11). These occur in a set of equality constraints, so the feasible region of any problem instance may be nonconvex. Since all instances we have solved have multiple local solutions, the feasible regions of all these instances are nonconvex. The L_2 and $L_{\max}$ models also have multiple local optima due to the nonconvex objective (13) and the nonconvex constraints (14). In order to find many local solutions, hopefully including the global one, we employed a new multi-start global optimization procedure called OptQuest/NLP (OQNLP) (Ugray et al. 2001, 2005; Lasdon et al. 2005) to solve this problem. OQNLP is interfaced to the popular algebraic modeling language GAMS (see www.gams.com), and this language is well suited to closed-form models with summations over various subsets of the variable indices, so we expressed the problem (5), (8)–(12) in GAMS. Several of the example problems solved here are available on the GAMS Development Corporation's "Global World" website, in a set of problems called "Globallib" at http://www.gamsworld.org/global/globallib.htm.

The OQNLP algorithm is an effective heuristic which calls a local NLP solver from a set of diverse starting points. When it is called from GAMS, any GAMS NLP solver can be used. We have used the CONOPT solver (Drud 1994), because it is efficient for large problems, and it has quickly solved (i.e. found local optima for) many instances of this model with M

ranging from 8 to 21. For the larger problems described in Sect. 5.3 we replaced CONOPT by SNOPT, an implementation of the Successive Quadratic Programming algorithm (Gill et al. 2005). The starting points for the NLP Solver are generated by a search algorithm called OptQuest (Laguna 1997), which maintains a population of points (default size 10). This is updated periodically, and new trial points are created as linear combinations of pairs of population points. The algorithm has 2 stages: in stage 1 (default length 200 trial points), the objective and constraint values at trial points are computed, no Solver calls are made, and the best trial point found is chosen to start stage 2. In that stage (default length 800 trial points), the first iteration calls the NLP Solver at the best stage one point, but the Solver is called at subsequent trial points if and only if two "filtering" tests are passed. These are described in (Ugray et al. 2005), and improved versions are in (Lasdon et al. 2005). We provide a brief pseudo-code description of this algorithm below. See (Ugray et al. 2001; 2005; Lasdon et al. 2005) for more complete descriptions.

Although OQNLP can be applied to problems with both continuous and discrete variables, the problems considered here have only continuous variables. Thus we restrict our attention to optimization problems having the form:

$$\text{Minimize } f(x) \tag{20}$$

subject to the general constraints

$$l \leq G(x) \leq u \tag{21}$$

and the bound constraints

$$x \in S \tag{22}$$

where x is an n-dimensional vector of continuous decision variables, G is an m-dimensional vector of constraint functions, and the vectors u and l contain upper and lower bounds for these functions. The set S is defined by simple bounds on x, and we assume that it is closed and bounded, i.e., that each component of x has a finite upper and lower bound. This is required by the starting point generators. The objective function f and the constraint functions G are assumed to have continuous first partial derivatives at all points in S. This is necessary so that a gradient-based local NLP solver can be applied to the NLP (20)–(22).

The *L1* exact penalty function is used as a merit function for evaluating candidate starting points. For the problem (20)–(22) this function is

$$P(x, w) = f(x) + \sum_{i=1}^{m} w_i \, viol(g_i(x)) \tag{23}$$

where the w_i are positive penalty weights, $g_i(x)$ is the *ith* component of G(x), and the function $viol(g_i(x))$ equals the absolute violation of the *ith* constraint of (2) at the point x. No terms are included for bound violations because the bounds are always satisfied by the algorithms considered here. It is well known (Nash and Sofer 1996) that if x^* is a local optimum of (19)–(21), λ^* is a corresponding optimal Lagrange multiplier vector, the second order sufficiency conditions are satisfied at (x^*, λ^*), and

$$w_i > abs(\lambda_i^*) \tag{24}$$

then x^* is a local unconstrained minimum of P. If (19)–(21) has several local minima, and each w_i is larger than the maximum of all absolute Lagrange multipliers for constraint i over all these optima, then P has a local minimum at each of these local constrained minima.

Let NEXT_CANDIDATE(xt) denote the starting point generator or driver, where xt is the candidate starting point produced by that procedure and $xt(i)$ refers to the ith candidate point generated. We refer to the local NLP solver as $L(xs, xf)$, where xs is the starting point and xf the final point. Pseudo-code for the algorithm, ignoring initializations, scalar input arguments, and other details, is given below. In it, the function UPDATE LOCALS(xs, xf, w, R) processes and stores solver output xf, produces updated penalty weights, w, and updates the radii of the basins of attraction of all local solutions found thus far. R is the vector of all these basin radii. A new point xf found by the NLP solver is stored in a list of "distinct" local optima, henceforth referred to as the "list of locals". Two local solutions $xf1$ and$xf2$ are considered distinct if $\|xf1 - xf2\| \leq epsl$ where $\|.\|$ denotes the infinity norm and the default value of $epsl$ is 0.001.

After an initial call to L at the user-provided initial point, x_0, stage 1 of the algorithm performs $n1$ iterations in which NEXT_CANDIDATE*(xt)* is called, and the *L1* Exact penalty value $P(xt, w)$ is calculated. The point with the smallest of these P values, denoted xt^* below, is chosen as the starting point for the next call to L, which begins stage 2.

OQNLP Algorithm Pseudo-code

STAGE 1

x_0 = user initial point
Call $L(x_0, xf)$ *attempt to generate initial local solution at user initial point*
Call UPDATE LOCALS(x_0, xf, w, R)
For *n1* candidate points, indexed by i: *generate Stage 1 set of candidate points*
    Call NEXT_CANDIDATE(*xt*(i))
    Evaluate $P(xt(i), w)$
$xt^* = \underset{1 \leq i \leq n1}{\text{argmin}}\, P(xt(i), w)$ *select candidate point with best penalty function value*
Call $L(xt^*, xf)$ *Call local solver at best stage 1 point*
Call UPDATE LOCALS(xt^*, xf, w, R)
threshold = $P(xt^*, w)$ *initialize merit filter threshold*

STAGE 2

For *n2* candidate points, indexed by i: *iterate for n2 candidate points*
    Call NEXT_CANDIDATE($xt(n1 + i)$)
    Evaluate P($xt(n1 + i), w$)
    Perform merit and distance filter tests:
    dstatus = distance filter($xt(n1 + i)$) *result is accept or reject candidate point*
    mstatus = merit filter($xt(n1 + i)$, threshold) *result is accept or reject*
    IF (dstatus and mstatus = "accept")
        Call $L(xt(n1 + i), xf)$
        Call UPDATE LOCALS($xt(n1 + i), xf, w, R$)

In the logic above, feasible points returned by the solver are inserted in the list of locals even if the point does not satisfy the Kuhn–Tucker conditions to within the solver tolerances. Such points may still have the best objective value found. The penalty weight update insures that w_i is always larger than the largest absolute multiplier for constraint i over all local optima.

Since our goal is to find all local solutions of this problem, we disabled the filtering tests in stage 2 of OQNLP. This causes the NLP Solver to be called at every stage 2 iteration. The default lengths for stages one and 2 given above were used, but we found that after 50 to 100

solver calls, no improved local solution was found. Hence we employed an OQNLP option which terminates the algorithm when 150 consecutive solver calls fail to locate an improved objective value. This usually led to runs of 500 iterations or less, which took about 20-30 seconds on a 1 Ghz Pentium 4 PC. The default lengths for stages one and two were used for all examples, because we had found previously that these values are very effective over a wide range of problem sizes-see Ugray et al. (2001, 2005), and Lasdon et al. (2005). While increasing both values is often useful in obtaining higher accuracy, this was not done here because many local solutions with small positive objective values were obtained using the default settings. Similarly, experiments with more than 150 consecutive solver calls generally led to little or no improvement in the set of local solutions found.

5 Computational results

In this section, we provide computational results from a series of experiments on four different populations. Three were artificial and one was from real data. The first population considered is P_2, an artificial population with 1000 classes of size 1 and 900 of size 10. Its class size distribution is very irregular, and it has been difficult (Greenberg and Zayatz 1992) to estimate the number of uniques in the population. Two additional artificial populations were used for testing. Unlike P_2, these populations have smooth class size distributions. P_4 is uniformly distributed with 100 classes of each size ranging from size 1 to size 14. P_5 is similar to a normal distribution with classes sizes increasing from 9 classes of size 1 to 90 classes of size 10 and then decreasing back down to 9 classes of size 21. In all computations involving these populations, the CONOPT local NLP solver was used.

One additional population, P_3, was considered for testing. This was a real population taken from census data found in Zayatz (1991). P_3 is much larger than the others, with 56,372 individuals. Of these, 22,026 were classified as unique. Other class sizes varied, with the largest having 298 individuals. Zayatz (1991) provided a sample from P_3. All other samples used in our testing were simulated.

5.1 Estimates of irregular class size distributions (P_2)

Tables 1–4 below show the estimates $\{\hat{\mu}_i\}$ obtained by the recursive and optimization approaches using the L_1 norm for 50%, 30%, 20%, and 10% samples from a population P_2 with 1000 classes of size 1 and 900 of size 10. In Tables 2–5, all local solutions with small norms are shown, and all class size values are rounded to integers. The best L_1 solution and the recursive procedure produce very similar results for the 50% and 30% samples from P_2, and closely estimate the class size distribution of the population.

The recursive and best optimization estimates are not as accurate or similar to one another for the 20% sample (Table 3), but they both roughly identify the population. However, the number of local L_1 solutions grows as the sampling rate decreases, and the third best solution for the 20% sample is quite different from the best. It has no uniques, many classes of size 7 and 9, and represents an alternative population whose expected sample is close to the input sample.

For the 10% sample from P_2 (Table 4), the recursive method converges to two size distributions from 2 different starting points, one correct and one with no small and many large classes. The OQNLP algorithm finds 4 low norm local L_1 solutions for this sample, with the first and third best similar to the true population, but the other two sharply different. Solutions 1 and 3 have many uniques and many large classes, while 2 and 4 have no classes of sizes 1 through 6 and many of sizes 7 through 10. The “standard” and “shift” columns of

Table 7 Actual sample and expected samples from L_1 solutions-10% sample from P_2

Sample	Error	Class sizes									
		1	2	3	4	5	6	7	8	9	10
Actual		449	174	52	10	1	0	0	0	0	0
Expected $L_{1,1}$	0.0033	449	174.7	51.0	9.78	1.28	0.116	0.0073	0.0003	0	0
Expected $L_{1,2}$	0.0384	441.7	187.0	47.8	8.24	1.00	0.087	0.0053	0.0002	0	0
Expect $L_{1,3}$	0.0045	448.8	175.0	50.7	9.83	1.30	0.119	0.0076	0.0003	0	0
Expected $L_{1,4}$	0.0464	439.8	189.8	47.7	7.69	0.82	0.058	0.0027	0.0001	0	0

Tables 3 and 4 contain the norms described in Section 3.5 with both normalized by dividing by the norm of the true size distribution. These norms provide the same ranking of solutions for the 20% sample, and nearly the same for the 10%.

Although the two pairs of similar solutions in Table 4 are very different from each other, all are populations that could have generated the 10% sample. To see this, we calculate the "expected sample" for each solution using equation (10). Table 7 illustrates that for the four L_1 solutions listed in Table 4, the expected samples are all very close. The relative absolute error column gives the difference between expected and actual samples, $\sum_j \left|\mathrm{E}m_j - m_j\right| / \sum_j m_j$. This relative error is always small, with values from .0033 to .046. This is to be expected since (11) shows that the optimization model will try to make $m_j / \mathrm{E}m_j$ as close to 1 as possible.

Table 5 shows the estimates obtained by the recursive and optimization methods for a 10% sample from P_1, a population with no small and many large classes. The sample is very similar to the 10% sample from P_2 in Table 4. The estimates of both the recursive procedure and the best optimization solution are basically correct. Table 5 also shows the class size distributions and objective values of the best 5 (of 15 found) local solutions to the optimization problem, plus the worst. All have small objective values, ranging from 3.5 E−5 to 1.8E−4. The class size distributions are all skewed heavily toward the largest sizes: sizes 1 through 4 are mostly zero with occasional small positive values, while size 9 is by far the largest, followed by size 8.

Table 8 shows estimates based on four different 10% samples from P_2, corresponding to the best local solutions to the optimization problem for each sample. The results show that the class size distributions of these solutions are not all close to the actual P_2 distribution, but instead contain diverse distributions that could have generated the sample. The standard and shift columns contain the error norms described in Sect. 3.5. For the first sample, the $L_{1,4}$ solution is the only one closely resembling the true population, and it has the lowest value of both error norms by far. The second sample has two solutions (1 and 3) that resemble population P_2. The best of these, solution 3, has the smallest values of both error norms, but the L_1 error for solution 1 is about the same as for solutions 4 and 5, which have no small classes. The shift error norm clearly ranks solution 1 as second best. For samples 3 and 4, none of the local solutions resemble the true population, since they all estimate μ_1 through μ_6 to be zero.

5.2 Estimates of smooth class size distributions (P_4 and P_5)

Table 9 shows the 3 lowest L_1, L_2, and $L_{\max}$ norm class size distribution estimates for a population P_4 with 100 classes of each size ($\mu_1 = \mu_2 = \cdots = \mu_{14} = 100$), based on the 30% sample shown. No smoothing constraints (see Sect. 3.4) were included. The best

Table 8 Local solutions and error norms for three different 10% Samples from P_2

	obj	Class sizes										Norms	
		1	2	3	4	5	6	7	8	9	10	standard	shift
Population		1000	0	0	0	0	0	0	0	0	900		
Sample 1		440	171	56	10	2							
$L_{1,1}$	0.03	0	0	0	0	0	0	0	0	1041	63	1.24	0.957
$L_{1,2}$	0.12	0	0	0	0	0	0	273	197	0	652	0.81	0.863
$L_{1,3}$	0.16	0	0	0	0	0	0	0	564	0	549	0.89	0.923
$L_{1,4}$	1.03	760	0	0	0	0	0	0	0	0	924	0.18	0.114
Sample 2		440	186	42	13	2							
$L_{1,1}$	2.9E−4	0	468	0	0	0	0	0	0	135	785	0.83	0.563
$L_{1,2}$	2.7E−3	0	0	0	0	0	0	0	87	1033	0	1.26	0.989
$L_{1,3}$	1.7E−2	578	0	0	0	0	0	0	209	46	733	0.37	0.383
$L_{1,4}$	1.1E−1	0	0	0	0	0	0	328	204	0	607	0.83	0.958
$L_{1,5}$	1.4E−1	0	0	0	0	0	0	481	0	0	663	0.84	0.985
$L_{1,6}$	1.7E−1	0	0	0	0	0	0	0	647	0	483	0.94	1.009
Sample 3		448	151	64	13	1							
$L_{1,1}$	0.09	0	0	0	0	0	0	0	0	979	119	1.19	0.925
$L_{1,2}$	0.21	0	0	0	0	0	0	241	208	0	666	0.80	0.824
$L_{1,3}$	0.25	0	0	0	0	0	0	0	530	0	576	0.87	0.886
Sample 4		440	171	56	10	2							
$L_{1,1}$	0.03	0	0	0	0	0	0	0	0	1041	63	1.24	0.957
$L_{1,2}$	0.12	0	0	0	0	0	0	273	197	0	652	0.81	0.863

Table 9 Nonsmooth estimates of Uniform 14 class population, 30% sample

	obj	Class sizes														Norm
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	standard
Population		100	100	100	100	100	100	100	100	100	100	100	100	100	100	
Sample		303	301	215	171	113	37	15	3	0	0	0	0	0	0	
$L_{1,1}$	2.72E−3	1	0	0	541	0	0	0	0	0	531	0	0	232	0	1.90
$L_{1,2}$	4.78E−3	0	43	0	521	0	0	0	0	0	371	0	385	0	0	1.76
$L_{1,3}$	5.36E−3	331	0	0	0	0	488	0	0	0	0	572	0	0	68	1.94
$L_{2,1}$	2.73E−6	0	0	0	541	0	0	0	0	0	531	0	0	232	0	1.90
$L_{2,2}$	1.19E−5	0	44	0	519	0	0	0	0	1	371	0	384	0	0	1.76
$L_{2,3}$	2.08E−5	331	0	0	0	0	488	0	0	0	0	571	1	1	67	1.94
$L_{\max,1}$	6.74E−4	0	0	0	541	0	0	0	0	0	531	0	0	232	0	1.90
$L_{\max,2}$	2.02E−3	330	0	0	0	0	487	0	0	0	0	571	1	1	67	1.94
$L_{\max,3}$	2.54E−3	0	1	432	0	0	102	4	4	341	0	0	453	1	0	1.67

L_1 estimates all have 4 positive components, all others zero, and this is typical of the 30 local optima found. The "standard" column contains the standard error norm with values ranging from 1.79 to 2.03 times the population norm. For the L_1 norm, all components of the errors, $pdev_i - ndev_i$, in the estimator equations (12) are zero except one.

The estimates produced using the L_2 and $L_{\max}$ objectives have similar properties, with 3 or 4 large components and the rest either much smaller or zero. The max norm estimates have a few large components, a few very small ones, with the rest zero. The same pattern occurs with a bell-shaped population distribution (P_5) with $M = 21$.

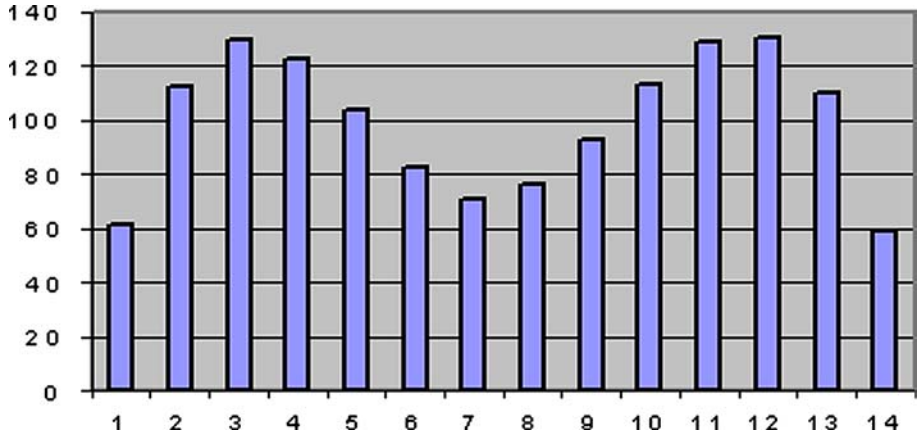

Fig. 2 Estimate from a 50% sample, Uniform population

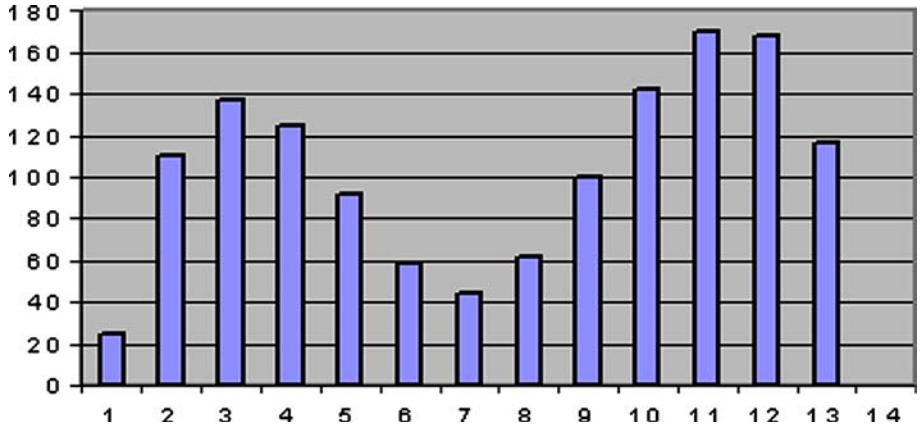

Fig. 3 Estimate from a 30% Sample, Uniform population

Figures 2–4 below plot the best L_1 solutions for 50%, 30%, and 10% samples from the uniform population described above, adding the smoothness constraints described in Sect. 3.4. These estimates are much better than those generated without the smoothness constraints, and are reasonably accurate for the 50% and 30% samples. The standard error norms, for these sampling rates are 0.25 and 0.51 respectively. The standard error norm for the best max norm local solution for the 10% sample is 1.90, but the second best local solution for this sample using the max norm has a standard error norm of 0.63, and is graphed in Fig. 5 below.

Figures 6–9 plot the estimates obtained using the L_1 norm with smoothing constraints, based on 50%, 30%, and 10% samples from population P_5 with a bell-shaped population distribution. These estimates are very accurate for the 50% sample, and moderately accurate for 30%. Of the two 10% estimates, the lowest L_1 norm estimate is quite inaccurate, with relative error 0.67, but the second best solution has relative error of only 0.14, demonstrating again the value of obtaining multiple local solutions.

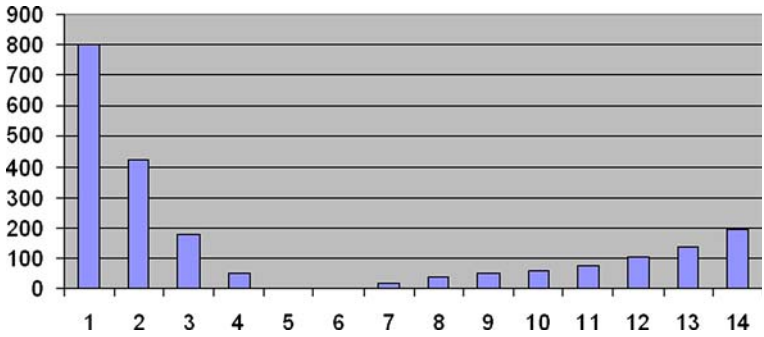

Fig. 4 Estimate from 10% sample, Uniform population

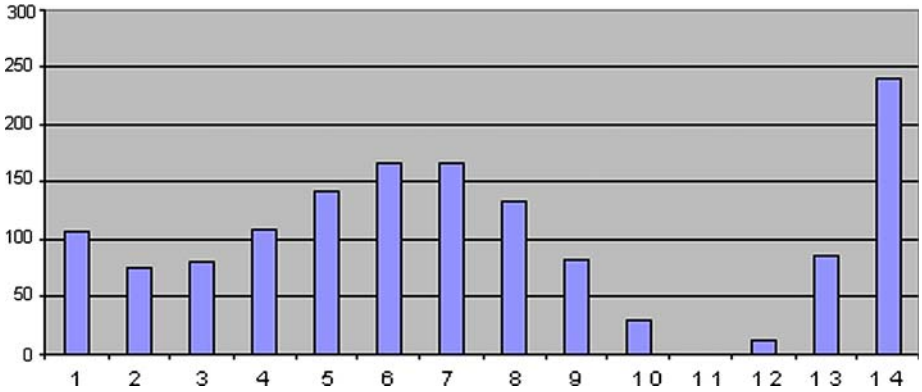

Fig. 5 Second best estimate, max norm, 10% sample, Uniform population

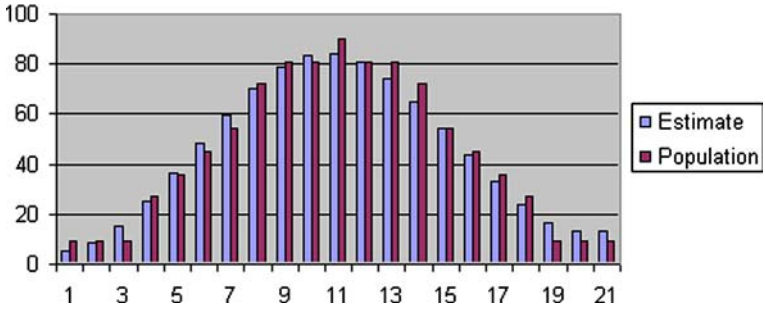

Fig. 6 Best L_1 estimate for 50% sample from bell-shaped distribution

5.3 Estimates for a large, realistic example

Table 10 illustrates results from 5 samples from a population we refer to as P_3. This is a large, realistic population based on census data with 56, 372 individuals and 22026 uniques.

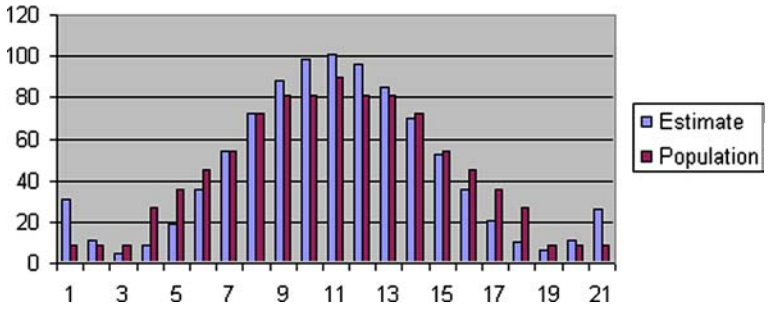

Fig. 7 Best L_1 estimate for 30% sample from bell-shaped distribution

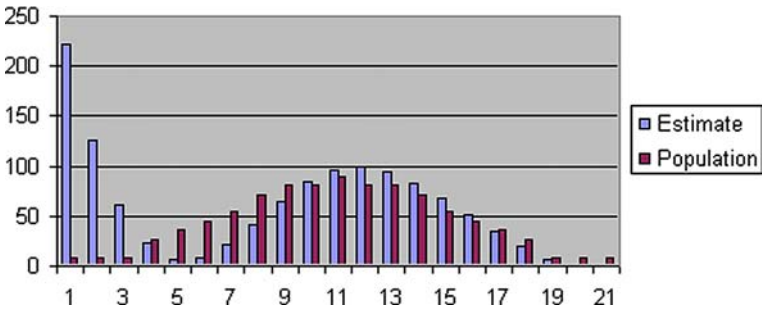

Fig. 8 Best L_1 estimate for 10% sample from bell-shaped distribution

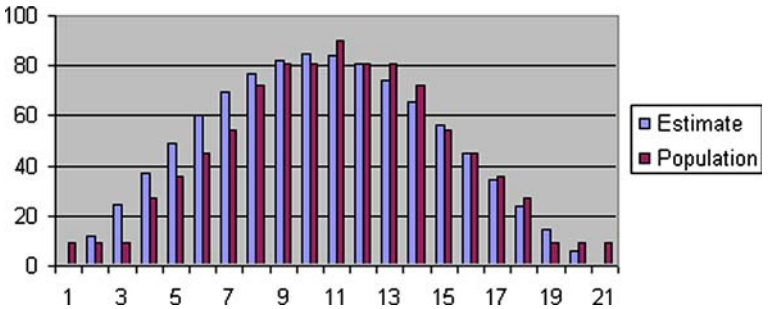

Fig. 9 Second best L_1 estimate for 10% sample from bell-shaped distribution

The samples are each of size 9,383, representing the 1 in 6 sampling rate which has been used for the Census long form. We found this practical problem to be much more difficult to solve than the artificial problems considered above. Solution times were long and many poor solutions were generated. To deal with this, Steel (1999) suggested that a monotonicity constraint $\mu_i \geq \mu_{i+1}, i = 1, \ldots, M-1$ would be appropriate. This added constraint was used to correct for choppiness and to reflect the belief that the populations with disclosure risk have a high number of uniques, fewer pairs, etc. This adds an additional $M-1$ constraints to the optimization problem, but these do not increase the solution time of either CONOPT or SNOPT (they may decrease it), and adding them significantly improved the results. To illustrate the problem sizes associated with our 5 samples from P_3, the optimization problem

Table 10 Results for real population P_3 (correct number of uniques = 22026)

Sample	Estimate associated with best objective	Relative error (%)	Mean of estimates of all solutions	Relative error (%)	Median of estimates of all solutions	Relative error (%)	Maximum of estimates of all solutions	Relative error (%)	Number of local solutions obtained
1	15379	30	18355	17	19056	13	21674	2	13
2	18701	15	19903	10	18717	15	22279	1	9
3	23983	9	22781	3	23983	9	23984	9	8
4	23746	8	23007	4	23786	8	25194	14	13
5	25179	14	23705	8	24961	13	25235	15	26

for sample 2 has 641 constraints and 917 variables. Running the 5 samples with a limit of 30 SNOPT calls led to solution times of from 3 to 5 minutes.

We found that when CONOPT was used as OQNLP's local Solver, many CONOPT calls terminated without finding a feasible solution. CONOPT first minimizes the sum of constraint violations in an attempt to find a feasible solution, then minimizes the true objective from that point. In problem instances associated with this large population, CONOPT often ended with an infeasible local minimum of the sum of constraint violations. Our prior experience was that SNOPT (Gill et al. 2005) often has fewer infeasible terminations in such situations, and that proved true again. Thus these experiments used SNOPT.

When multiple solutions are generated, it is not clear how to use the results to estimate disclosure risk. One possibility is to be conservative and use the solution with the largest value for the number of uniques. Other possibilities are to use the mean or median of all the solutions obtained. The mean, median, and maximum of the estimate of uniques over all local solutions obtained were calculated for each of 5 samples from P_3. Estimates of the number of uniques for the solution with the best objective value, and the mean, median, and maximum of this estimate over all solutions obtained, plus the number of local solution used for these calculations is listed in Table 10. Relative errors of each estimation procedure are also shown. As expected, the solution with the lowest objective value is usually not the one that is closest to the correct value. Overall, it appears to be best to use the solution with the largest number of uniques to estimate disclosure risk. This has the advantage of being conservative and relatively accurate. More experiments are needed to assess the generality of this outcome.

6 Conclusions

Formulating this difficult class of estimation problems as the minimum norm solution of a system of nonlinear equations leads to a nonconvex problem which often has many local solutions, especially for low sampling rates. These solutions correspond to populations which could have generated the input sample. When more than one population can generate a sample, the OQNLP multistart solver finds multiple—if not all—local solutions.

Unless additional information is available, there is not likely to be a method that could accurately determine from which population the sample was actually selected. However, when all potential solutions are identified, rather than just one that may be incorrect, disclosure risk can be more accurately assessed. Similarly, when multiple solutions are found

for the number of species in a population, this provides information about the degree of uncertainty in the estimates.

References

Bethlehem, J.G., Keller, W.J., Pannekoek, J.: Disclosure control for microdata. J. Am. Stat. Assoc. **85**, 38–45 (1990)

Bunge, J., Fitzpatrick, M.: Estimating the number of species: A review. J. Am. Stat. Assoc. **88**, 364–373 (1993)

Chen, G., Keller-McNulty, S.: Estimation of identification disclosure risk in microdata. J. Official Stat. **14**, 79–95 (1998)

Dalenius, T.: A Simple Procedure for Controlled Rounding. Norstedts Tryckeri, Stockholm (1981)

Dalenius, T., Reiss, S.P.: Data swapping: A technique for disclosure control. J. Stat. Plan. Infer. **6**, 73–85 (1982)

De Waal, A.G., Willenborg, L.C.R.J.: Optimal local supression in microdata. J. Official Stat. **14**, 421–435 (1998)

Drud, A.: CONOPT — A Large Scale GRG Code, ORSA J. Comput. **6**, 207–216 (1994)

Efron, B., Thisted, R.: Estimating the number of unseen species: How many words did Shakespear know? Biometrika **63**, 435–447 (1976)

Gill, P.E., Murray, W., Saunders, M.A.: Users Guide for SNOPT Version 7, Department of Management Science and Engineering, Systems Optimization Laboratory, Stanford University, Stanford, CA, 94305-4026, USA, March 20, (2006)

Greenberg, B.G., Zayatz, L.V.: Measuring risk in public use microdata files, Statistica Neerlandica **46**, 33–48 (1992)

Greenberg, B.S.: New Approaches to Estimate Disclosure Risk, Presented at the NSF Confidentiality Workshop, Washington, DC, May 12–13 (2003). Retrieved June 1, 2005 from http://www.urban.org/nsfpresentations/pdfs/05_Greenberg.pdf

Haas, P., Naughton, J., Sehadri, S., Stokes, L.: Sampling-based estimation of the number of distinct values of an attribute. VLDB 95: Proceedings of the International Conference on Very large Databases (In: Dayal, U., Gray, P., Nishio, S. (eds.) pp. 311–322 (1995).

Hoshino, N.: Applying Pittman's sampling formula to microdata disclosure risk assessment. *J. Official Stat.*, **17**, 499–520 (2001)

Kim, J.: A method for limiting disclosure in microdata based on random noise and transformation. Proceedings of the Section on Survey Research Methods Section. American Statistical Association, Alexandria, VA pp. 370–374 (1986)

Laguna, M.: Optimization of Complex Systems for OptQuest (1997). Retrieved May 23, 2005 from http://www.crystalball.com/optquest/complexsystems.html

Lasdon, L., Plummer, J., Ugray, Z., Bussieck, M.: Improved filters and randomized drivers for multi-start global optimization. Submitted to *Journal of Global Optimization*, March 2005

Madigan, D., York, J.C.: Bayesian methods for estimation of the size of a closed population. Biometrika **84** (1), 19–31 (1997)

Nash, S.G., Sofer, A.: (1996), Linear and Nonlinear Programming, McGraw-Hill, New York

Skinner, C.J., Holmes, D.J.: Modelling population uniqueness. Proceedings of the International Seminar on Statistical Confidentiality. pp. 175–199. Statistical Office of the European Communities, Luxembourg, (1993)

Smith-Cayama, R.A., Thomas, D.R.: Estimating the number of distinct valid signatures in initiative petitions. Proceedings of the Survey Research Methods Section. pp. 238–243. American Statistical Association, Alexandria, VA, (1999)

Takemura, A. Some superpopulation models for estimating the number of population uniques. Statistical Data Protection — Proceedings of the Conference, Lisbon, 25–27 March 1998–1999 edition, pp. 59–76. Office for Official Publications of the European Communities, Luxembourg (1999)

Ugray, Z., Plummer, J.C., Glover, F.W., Kelly, J., Lasdon, L.S., Marti, R.: A multistart scatter search heuristic for smooth NLP and MINLP problems. Conference on Adaptive Memory and Evolution: Tabu Search and Scatter Search. University of Mississippi at Oxford, March 8–10, (2001)

Ugray, Z., Plummer, J.C., Glover, F.W., Kelly, J., Marti, R.: Scatter search and local NLP solvers: A multistart framework for global optimization. To appear in *INFORMS Journal on Computing*.

White, J.K., Sangiovanni-vincentelli, A.: Relaxation Techniques for the Simulation of VLSI Circuits, Kluwer Academic Publishers (1987)

Willenborg, L., DeWaal, T.: Elements of Statistical Disclosure Control. Springer-Verlag, New York (2001)

Zayatz, L.V.: Estimation of the percent of unique population elements in microdata file using the sample. Statistical Research Division Report Series, Census/SRD/RR-91/08 (1991).