

Modernization of Benchmarking Economic Time Series at the U.S. Census Bureau

Irene Brown

U.S. Census Bureau
4600 Silver Hill Road, Washington, DC 20233 / Irene.Brown@census.gov

Abstract

A common problem faced by government agencies that collect and publish time series data is to maintain a consistent set of time series. Generally, this is done by adjusting monthly or quarterly series to match the less frequent series (annual or Census) that are assumed to be of higher quality, while simultaneously preserving some characteristic of the original more frequent series. Benchmarking refers to techniques used to solve this problem of incoherence between time series data measured at different frequencies.

Economic Programs at the Census Bureau use the Causey-Trager procedure for benchmarking monthly and quarterly series to annual series and to the Economic Census every five years. This procedure, developed thirty years ago, uses an iterative, nonlinear technique known as steepest (gradient) feasible descent to obtain the benchmarked series by optimizing a measure of growth-rate preservation.

Even though the software implementation of the procedure has changed a little with each new programming language, the methodology has not changed in the last three decades. This paper describes the Causey-Trager benchmarking methodology along with current and future research to improve the benchmarking methods of economic time series at the U.S. Census Bureau.

1. Introduction

The U.S. Census Bureau's Economic Programs (<http://www.census.gov/econ/progoverview.html>) provide statistics about U.S. businesses and government organizations. These statistics are obtained from an Economic Census and Census of Governments every five years and from over 100 separate surveys taken monthly, quarterly, and annually including twelve principal economic indicators. Many of these surveys have the same target population but data is gathered at different frequencies (e.g. Monthly Retail Trade Survey and Annual Retail Trade Survey).

Gathering data at different frequencies balances the need for comprehensive, reliable, detailed statistics versus the need for timeliness. The Economic Census and annual surveys provide more reliable detailed level estimates, while the monthly and quarterly surveys provide more timely estimates of trend (or period-to-period changes). Annual surveys have larger sample sizes, and more time to process, edit, and reduce non-response. Where, surveys that gather data more frequently are focused on obtaining better estimates of period-to-period changes than the level estimates. Because of survey error, the sets of time series for this same target population are not consistent. For example, the sum of the monthly sales values usually does not equal the sales value from the annual survey. Benchmarking combines the information from both surveys into a consistent higher frequency series by preserving some characteristic of the higher frequency series while attaining the levels of the less frequent series. Figure 1 graphically illustrates this benchmarking concept.

Since the main purpose of the higher frequency series is to track the trend, the Census Bureau's benchmarking goal is to preserve the period-to-period changes (or growth rates) of the higher frequency series. Causey and Trager (1981; see also Trager, 1982, and Bozik and Otto, 1988) developed a method as the solution to this problem. Even though the software implementation of the procedure has changed programming languages from ALGOL to ASCII Fortran to SAS[®], the basic methodology remained the same.

This paper first gives some basics and notations of benchmarking. Then, it describes the Causey-Trager solutions, along with a comparison to one of Denton's (1971) movement preservation methods. It also describes the

differences found between the previous software in Fortran with the newest software implementation in SAS, which has led to current and future research to improve the benchmarking methods at the Census Bureau.

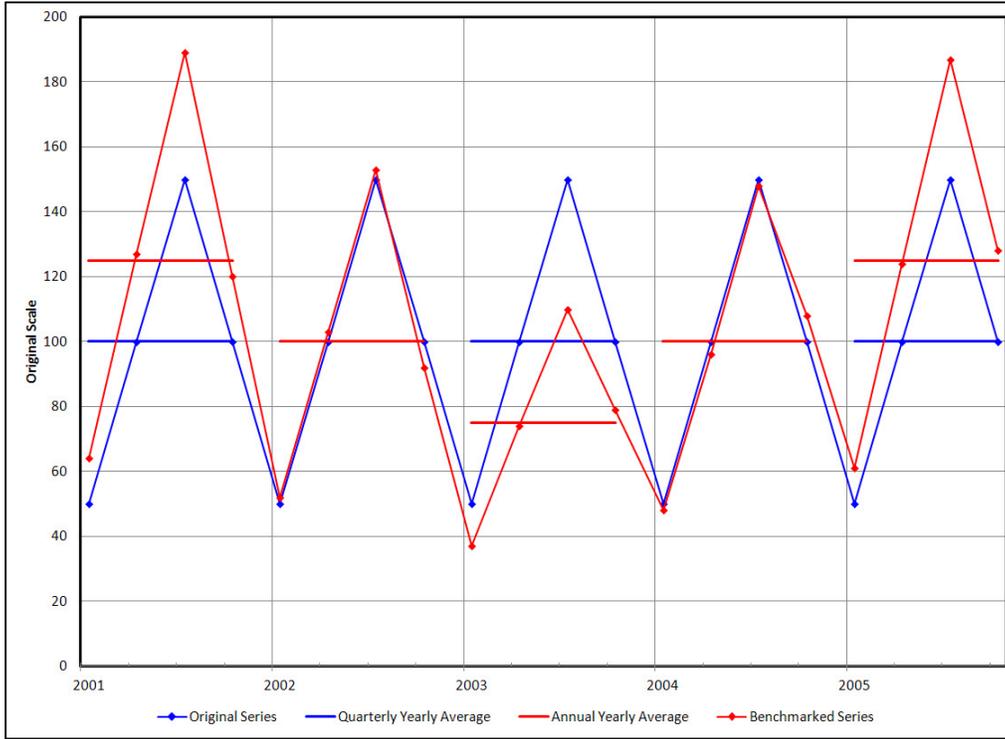


Figure 1: Time Series Plot of an Example Original Series and Benchmarked Series

2. Benchmarking Times Series

Common applications of benchmarking within the U.S. Census Bureau's Economic Programs are adjusting monthly or quarterly survey estimates to annual estimates and adjusting inter-censal annual figures to census figures. Let the higher frequency series be defined as the original series and denoted as:

$$x_t, t = 1, 2, \dots, n \quad (1)$$

Define the less frequent series as the benchmarks and denoted as:

$$T_k, k = 1, 2, \dots, m \quad (2)$$

The indices t and k represent time periods that map to specific dates in the series. Each benchmark covers a span of time periods from the original series, where b_k is the beginning time period for the k th benchmark and e_k is the ending time period for the k th benchmark such that

$$t = 1 \leq b_1 \leq e_1 < b_2 \leq e_2 < \dots < b_m \leq e_m \leq n$$

for $k = 1, 2, \dots, m$ non-overlapping benchmark periods.

The unknown target benchmarked series defined as the revised series and denoted as:

$$y_t, t = 1, 2, \dots, n \quad (3)$$

must meet the benchmark constraints such that

$$\sum_{t=b_k}^{e_k} y_t = T_k, k = 1, \dots, m \quad (4)$$

For Economic Programs at the U.S. Census Bureau, there are generally two types of series that are benchmarked, flow or stock, and their benchmark spans are defined differently. Flow series measure a variable over a time interval (e.g. sales) and therefore the benchmarks are the sum of the values over the benchmark period (i.e. $b_k < e_k$). Stock series measure the level of a variable at a point in time (e.g. inventories) and therefore the benchmarks are not the sum but a value at a specific time period (i.e. $b_k = e_k$). However, when benchmarking annual figures to the census figures, all series types behave like stock series in that $b_k = e_k$, regardless of the series being flow or stock. For example, the sales value for the 2007 Census is equal to the 2007 annual value not the sum of sales from 2003 to 2007.

There are many possible revised series (3) when meeting the benchmarking constraints (4) alone. Therefore, benchmarking methods must also try to preserve some characteristic of the original series. The simple procedure of pro-rating preserves the period-to-period changes exactly within each benchmark period, but greatly distorts the changes for the time period between benchmark periods. Table 1 shows this difficulty of pro-rating using Denton's (1971) illustrative example series. All growth rates of the revised series are exactly equal to the original series, except from 4th quarter to 1st quarter in all years. The majority of the benchmarking methods, therefore, select an object (or penalty) function that is minimized under some constraints of the original series and the new revised series to smooth out the discontinuities and avoid this issue.

Table 1: Denton's Quarterly Series Example for Pro-Rating

Time Periods		Benchmarks	Original Series	Revised Series (Pro-Rating)	Original Series Growth Rates (%)	Revised Series Growth Rates (%)
Year 1 -	1Q	500	50	62.5	-	-
	2Q		100	125	100	100
	3Q		150	187.5	50	50
	4Q		100	125	-33	-33
Year 2 -	1Q	400	50	50	-50	-60
	2Q		100	100	100	100
	3Q		150	150	50	50
	4Q		100	100	-33	-33
Year 3 -	1Q	300	50	37.5	-50	-62.5
	2Q		100	75	100	100
	3Q		150	112.5	50	50
	4Q		100	75	-33	-33
Year 4 -	1Q	400	50	50	-50	-33
	2Q		100	100	100	100
	3Q		150	150	50	50
	4Q		100	100	-33	-33
Year 5 -	1Q	500	50	62.5	-50	-37.5
	2Q		100	125	100	100
	3Q		150	187.5	50	50
	4Q		100	125	-33	-33

Denton (1971) developed a general solution to a class of variant object (penalty) functions based on the principle of movement preservation that are still widely used. He sets up a Lagrangian equation of the constrained optimization problem, takes partial derivatives and sets them equal to zero to create a system of linear equations to find the solution. All of his objective functions were linear, therefore a unique solution could be found to the system of linear equations. The Causey-Trager methods also use constrained optimization techniques to solve for two different objective functions to obtain a new revised series. One linear function being Denton's proportional first difference called the relative revision and another non-linear objective function called the trend revision. The two revisions, relative and trend, are described in sections 2.1 and 2.2.

2.1 Relative Revision

The relative revision minimizes the proportional first difference object function proposed by Denton:

$$F_R = \sum_{t=1}^{n-1} \left(\frac{y_{t+1}}{x_{t+1}} - \frac{y_t}{x_t} \right)^2 \quad (5)$$

given the benchmark constraints (4). Trager (1982) provided a solution using Lagrange multipliers similar to Denton's. The Lagrange equation is the same:

$$F'_R = \sum_{t=1}^{n-1} \left(\frac{y_{t+1}}{x_{t+1}} - \frac{y_t}{x_t} \right)^2 + 2 \sum_{k=1}^m \lambda_k \left(\sum_{t=b_k}^{e_k} y_t - T_k \right) \quad (6)$$

Both take the partial derivatives of equation (6) with respect to $y_t, t = 1, \dots, n$ and $\lambda_k, k = 1, \dots, m$ setting them equal to zero and creating a linear system of $n+m$ equations. With the solution in matrix form being:

$$\begin{bmatrix} \mathbf{y} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{B}' & \mathbf{0} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{0} \\ \mathbf{T} \end{bmatrix}$$

Where, \mathbf{y} is a $(n \times 1)$ vector of the revised values ($y_t, t = 1, \dots, n$),

$\boldsymbol{\lambda}$ is a $(m \times 1)$ vector of the Lagrange multipliers ($\lambda_k, k = 1, \dots, m$)

$\mathbf{A} = \mathbf{X}^{-1} \mathbf{D}' \mathbf{D} \mathbf{X}^{-1}$, $\mathbf{X} = \text{diag}(\mathbf{x})$,

\mathbf{x} is a $(n \times 1)$ vector of the original values ($x_t, t = 1, \dots, n$),

$$\mathbf{D} = \begin{bmatrix} -1 & 1 & 0 & 0 & \cdots \\ 0 & -1 & 1 & 0 & \cdots \\ 0 & 0 & -1 & 1 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix} \text{ a } ((n-1) \times n) \text{ first differences matrix,}$$

$\mathbf{B} = [\mathbf{b}_1 \quad \mathbf{b}_2 \quad \cdots \quad \mathbf{b}_m]$ a $(n \times m)$ temporal aggregation matrix, where \mathbf{b}_k is a $(n \times 1)$ vector of the elements equal to 1 for $t \in [b_k, e_k]$ and 0 for $t \notin [b_k, e_k]$, and

\mathbf{T} is a $(m \times 1)$ vector of the benchmark values ($T_k, k = 1, \dots, m$).

Because $\mathbf{D}' \mathbf{D}$ has rank $n-1$, \mathbf{A} is singular thus requiring the inversion of a possibly very large $(n+m \times n+m)$ matrix. Both Denton and Trager wanted to avoid this formidable task but took different approaches to the solution.

Denton set an initial condition of $y_0 = x_0$ and changed the first differences matrix to

$$\mathbf{D}_0 = \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots \\ -1 & 1 & 0 & 0 & \cdots \\ 0 & -1 & 1 & 0 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix},$$

a $(n \times n)$ matrix that is full rank thus making \mathbf{A} invertible. His solution uses a "well-known result for deriving the inverse of a partitioned matrix" (Denton 1971):

$$\mathbf{y} = \mathbf{x} + \mathbf{A}^{-1} \mathbf{B} (\mathbf{B}' \mathbf{A}^{-1} \mathbf{B})^{-1} (\mathbf{T} - \mathbf{B}' \mathbf{x}).$$

Denton said, "one of the attractive features of the foregoing approach is that the matrices have some convenient properties which make them easy to work with." (1971) This solution of the classical proportional Denton method

has at least one major shortcoming: “The method introduces a transient movement at the beginning of the series, which defeats the stated principle of movement preservation” (Dagum and Cholette 2006). Cholette’s (1984) modification of Denton’s first proportional difference restores the first difference matrix to its original form requiring the inversion of the larger matrix. Although Trager goes about solving it differently, the resulting revised series is the same as the modified proportional Denton.

Trager takes the steps of inverting the larger matrix and breaks them down into small pieces for the computer to compute. After taking the partial derivatives and setting them equal to zero, Trager adds an unknown constant (C) and reduces the system of equations down to $m+1$ linear equations and $m+1$ unknowns ($C, \lambda_1, \dots, \lambda_m$) as:

$$\sum_{k=1}^m \lambda_k \sum_{t=b_k}^{e_k} x_t = 0$$

and for $k = 1, 2, \dots, m$

$$- \sum_{r=1}^k \lambda_r \sum_{t=b_k}^{e_k} x_t \left(\sum_{p=b_r}^{\min(e_r, t-1)} (t-p)x_p \right) + C \sum_{t=b_k}^{e_k} x_t = T_k$$

The ASCII Fortran software implementation uses a modified Gram-Schmidt process to solve the smaller system of linear equations. The program written in SAS uses the INV function within the IML procedure to solve the system of equations, which uses the “LU decomposition followed by back substitution to solve for the inverse” (SAS 2008). Though the procedures are different, the actual resulting series are rarely different especially when rounded to the nearest integer as shown in Section 3.

Finally, to solve for the revised series, $C, \lambda_1, \dots, \lambda_m$ are substituted into the following equation:

For $t = b_1, \dots, e_m$

$$y_t = x_t C - x_t \sum_{k=1}^{\alpha_t} \lambda_k \sum_{j=b_k}^{e_k} (t-j)x_j - x_t \lambda_{\beta_t} \sum_{j=b_{\beta_t}}^{t-1} (t-j)x_j$$

where,

$$\alpha_t = \text{the maximum } k \text{ such that } t > e_k$$

$$\beta_t = \begin{cases} k, & \text{if } t \in [b_k, e_k] \\ 0, & \text{otherwise} \end{cases}$$

This differs slightly from Trager’s documented solution as the software implementation only solves for the time span covered by the benchmarks ($t \in [b_1, e_m]$). For those time periods outside the benchmarks, the software uses what is called a carry backward factor and a carry forward factor. These factors preserve the period-to-period changes exactly for those time periods outside the benchmarks. The factors, developed from the revised and original series, give an implicit projection for the unknown benchmark values.

The carry backward factor defined as:

$$y_t = x_t \left(\frac{y_{b_1}}{x_{b_1}} \right), \text{ for } t = 1, \dots, b_1 - 1 \quad (7)$$

in addition, the carry forward factor defined as:

$$y_t = x_t \left(\frac{y_{e_m}}{x_{e_m}} \right), \text{ for } t = e_m + 1, \dots, n \quad (8)$$

2.2 Trend Revision

The trend revision attempts to minimize the exact growth rate preservation object function:

$$F_T = \sum_{t=1}^{n-1} \left(\frac{y_{t+1}}{y_t} - \frac{x_{t+1}}{x_t} \right)^2 \quad (9)$$

Because the function is non-linear, Causey and Trager (1981) used a nonlinear programming method called steepest feasible descent to find a solution. To find a local minimum of a function using the method of steepest descent, one takes steps proportional to the negative of the gradient of the function at a current feasible point. The procedure continues iteratively to find a feasible point sufficiently close to the minimum, which is determined by a pre-set criterion. In addition, most nonlinear iterative procedures set a maximum number of iterations to avoid continuing indefinitely if there is no solution.

The method of steepest feasible descent begins with a starting point $(y_t^{(0)}, t = 1, 2, \dots, n)$ that meets the benchmarking constraints (4). Causey and Trager used the relative revision solution as the starting point in their algorithm. Generally, this should be a good starting point as Dagum and Cholette (2006) claim, "Proportional benchmarking can be seen as an approximation of the preservation of period-to-period growth rate of the original series in the benchmarked series."

At each iteration, a new feasible point (series) is found by:

$$y_t^{(j)} = y_t^{(j-1)} + s^{(j)} d_t^{(j)}, t = 1, 2, \dots, n \quad (10)$$

where,

$$d_t^{(j)}, t = 1, 2, \dots, n \quad (11)$$

is the steepest descent direction projected onto the feasible region,

$$s^{(j)} \quad (12)$$

is a scalar that reduces the function along the direction vector, and j represents the iteration number.

To project the negative gradient $(-\nabla F_T(y_t^{(j-1)}, t = 1, 2, \dots, n))$ onto the feasible region and find the direction vector (11), Causey and Trager set up the following constrained optimization problem:

$$\begin{aligned} & \text{minimize } \sum_{t=1}^n a_t^{(j)} d_t^{(j)} \\ & \text{subject to: } \sum_{t=1}^n d_t^{(j)2} = 1 \\ & \text{and } \sum_{t=b_k}^{e_k} d_t^{(j)} = 0, k = 1, 2, \dots, m \end{aligned}$$

where, $a_t^{(j)}, t = 1, \dots, n$ is the negative gradient. The set of constraints ensure the resulting solution will continue to meet the desired set of original benchmarking constraints (4). Using the Lagrange equation:

$$\sum_{t=1}^n a_t^{(j)} d_t^{(j)} + \frac{\lambda^*}{2} \left(\sum_{t=1}^n d_t^{(j)2} - 1 \right) + \sum_{k=1}^m \lambda_k \sum_{t=b_k}^{e_k} d_t^{(j)}$$

Take partial derivatives with respect to $d_t^{(j)}; t = 1, \dots, n$, and $\lambda^*, \lambda_k; k = 1, \dots, m$ and set them equal to zero. Using substitution, the solutions are:

$$\lambda_k = \frac{-\sum_{t=b_k}^{e_k} a_t^{(j)}}{(e_k - b_k + 1)}, k = 1, \dots, m$$

$$\lambda^* = \left[\sum_{k=1}^{m-1} \sum_{t=e_k+1}^{b_k-1} a_t^{(j)^2} + \sum_{k=1}^m \sum_{t=b_k}^{e_k} (-a_t^{(j)} - \lambda_k)^2 \right]$$

$$d_t^{(j)} = \begin{cases} -a_t^{(j)} & \text{for } t \notin [b_k, e_k] \\ \lambda^* & \text{for } k=1, \dots, m, t=1, \dots, n \\ \frac{-a_t^{(j)} - \lambda_k}{\lambda^*} & \text{for } t \in [b_k, e_k] \end{cases}$$

Note that if $\lambda^* = 0$ there is no solution or direction vector, the procedure is stopped and the last iteration ($j-1$) is used as the solution.

Since function (9) is non-linear, a line search method is used to find the scalar (12) that minimizes the function:

$$F_T^* = \sum_{t=1}^{n-1} \left(\frac{y_{t+1}^{(j-1)} + s^{(j)} d_{t+1}^{(j)}}{y_t^{(j-1)} + s^{(j)} d_t^{(j)}} - \frac{x_{t+1}}{x_t} \right)^2 \quad (13)$$

To find the minimum, consider the Taylor expansion of the first derivative of equation (13) about a point u .

$$F_T^{*'}(s^{(j)}) = 0 = F_T^{*'}(u) + F_T^{*''}(u)(s^{(j)} - u) + \frac{F_T^{*'''}(u)}{2}(s^{(j)} - u)^2$$

The line search steps are:

1. Evaluate $F_T^{*'}(u)$, $F_T^{*''}(u)$, and $F_T^{*'''}(u)$ at $u = 0$
2. Using the quadratic equation:
 - a. If $(F_T^{*''}(u))^2 - 2F_T^{*'''}(u)F_T^{*'}(u) > 0$ then the roots are:

$$\frac{-F_T^{*'}(u) \pm \sqrt{(F_T^{*''}(u))^2 - 2F_T^{*'''}(u)F_T^{*'}(u)}}{F_T^{*''}(u)}$$

And $D = \min\{|root_1|, |root_2|\}$ (minimum absolute root)

- b. Else $D = \frac{-F_T^{*'}(u)}{F_T^{*''}(u)}$
3. Compute $s^{(j)} = \frac{D}{2}$

This methodology will not yield a valid solution if any of the following occur:

- 1) $s^{(j)} < 0$
- 2) $s^{(j)} > \min\left\{\frac{-y_t^{(j-1)}}{d_t^{(j)}}\right\}$, where $d_t^{(j)} < 0$
- 3) $F_T^{*'}(s^{(j)}) > 0$
- 4) $F_T^{*''}(s^{(j)}) < 0$

Once again, if any of the above is true the procedure is stopped and the last iteration ($j-1$) is used as the solution.

If both the direction vector and scalar are found, the next feasible point (equation (10)) is calculated and tested. The iterative procedure stops when either the stopping rule:

$$\frac{F_T(y_t^{(j-1)}, t=1, \dots, n)}{F_T(y_t^{(j)}, t=1, \dots, n)} \leq 1.00001 \quad (14)$$

occurs or the maximum number of iterations is reached. Once again, the ASCII Fortran program only uses the series span where benchmarks are available ($t \in [b_1, e_m]$) and applies carry backward/forward factors as shown in equations (7) and (8). This shortened span applies to all of the equations above as if

$$t = 1 = b_1 \leq e_1 < b_2 \leq e_2 < \dots < b_m \leq e_m = n .$$

2.3 Comparing Benchmarking Methods

Using the example series from Denton (1971), the revised series from the methods pro-rating, classical Denton, relative revision, and trend revision can be compared on several criteria. Table 2 shows the objective function values for the relative and trend revisions by benchmarking method. As expected, both the relative and trend revision achieved the minimum of their respective objective functions.

Table 2: Comparison of Objective Function Values by Benchmarking Method for Denton’s Example Series

Objective Function	Pro-Rating	Classical Denton	Relative	Trend
Relative	0.25000000	0.09865556	0.07974444	0.13473333
Trend	0.06902778	0.18432354	0.15623692	0.04604445

Figure 2 shows the ratio of revised series values to the original series values, the criterion that is minimized by the Denton or relative revision. Both of those methods produce smooth curves, where pro-rating has jumps at the benchmark periods, and the trend revision has smaller jumps but is not as smooth as the relative revision. Also, for the classical Denton method the first two points are vastly different from the other methods. Denton’s initial condition causes this spurious movement at the beginning.

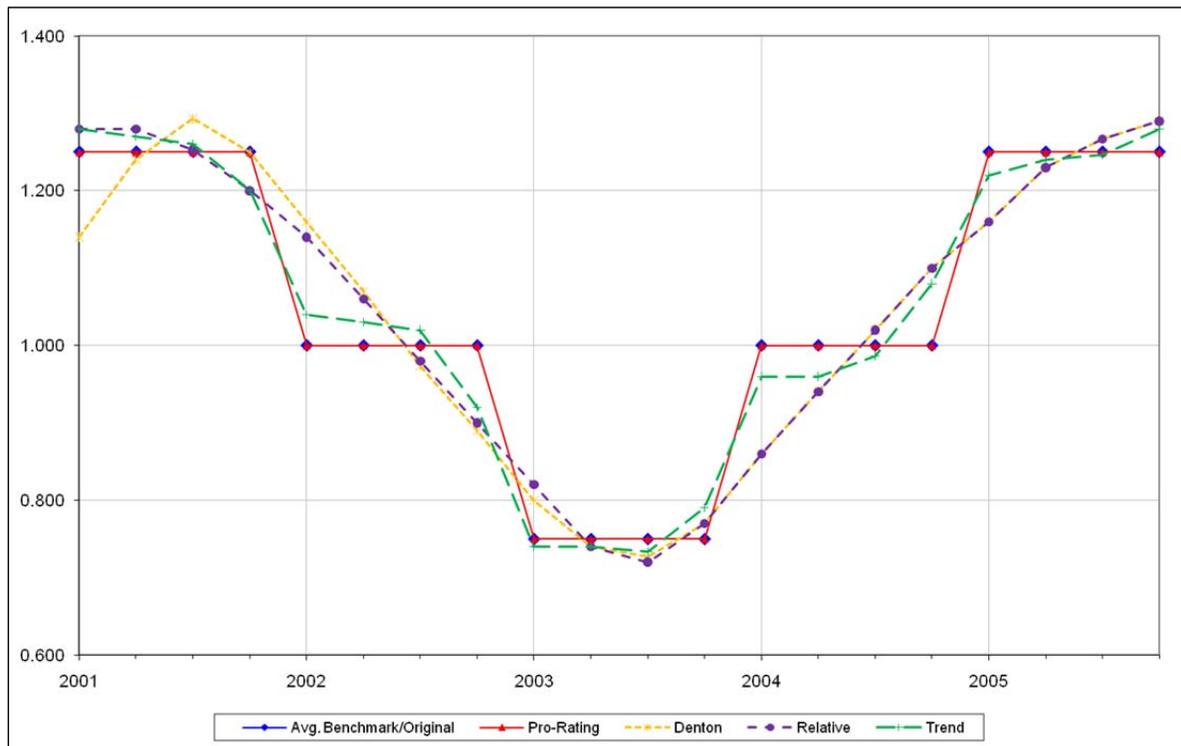


Figure 2: Ratio of Revised Series to Original Series by Method

The initial condition problems can also be seen in Figure 3, which shows the percent growth rates for the original series and revised series by method. The Denton method growth rates are very different from the original for the first two periods, an undesirable trait. The pro-rating method matches the original series on all quarters except between benchmark periods or years, where there are large differences, also undesirable. The relative revision has

difference for all the quarters and the trend revision tends to preserve the period-to-period changes at each period a little better overall.

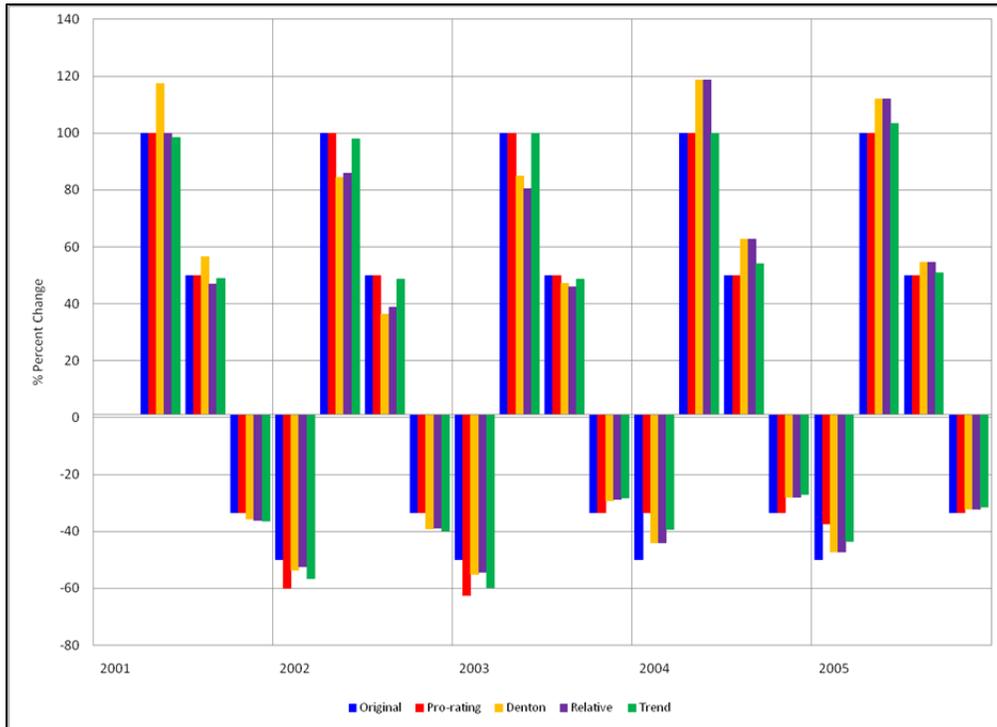


Figure 3: Bar Chart of Percent Growth Rates for Original Series and Revised Series by Method

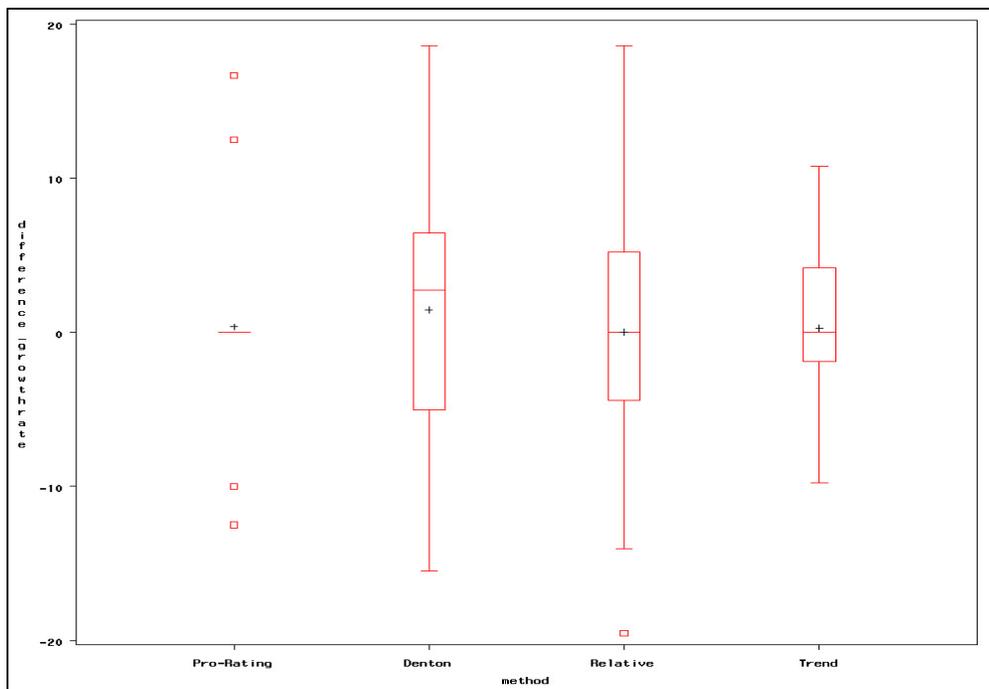


Figure 4: Comparison Box Plots of the Revisions in Growth Rates by Method

Figure 4, comparison box plots of the revisions in growth rates from the original series, also shows these patterns. Where the revisions are calculated as:

$$u_t^{Method} = \left(\frac{y_{t+1}^{Method}}{y_t^{Method}} - \frac{x_{t+1}}{x_t} \right) * 100 \text{ for } t = 1, \dots, n$$

As to which method produces, the “best” revised series depends on individual preferences for what criteria from the original series should be preserved. At the U.S. Census Bureau, preserving the period-to-period or growth rates ranks the highest. The differences between the methods also depend on certain characteristics of the original series and benchmarks, but the general pattern is still illustrated by this example.

3. Empirical Differences between Software Implementations

Although both programs, ASCII Fortran and SAS, used the same methodology, some differences exist in the resulting revised series. To understand the differences, currently benchmarked economic time series coming from the Manufacturing, Construction, Retail, Wholesale, and Services industries from the U.S. Census Bureau were used to compare the two programs. Of the 1455 series, 51 percent were annual series, 4 percent were quarterly (all flow series), and 45 percent were monthly (43 percent flow and 57 percent stock). For each original series, four revised series, one from each program (Fortran and SAS) and revision type (relative and trend), were created. Table 3 shows the percentage of series where the objective functions, number of iterations, and at least one time period values were different. For the relative revision, there were no differences in the objective function (rounded at the 8th decimal place) and in the revised series when rounded to the nearest integer.

Table 3: Percentage of Series that are Different between the Fortran and SAS Programs

				Percentage of Series Different		
Revision	Frequency	Series Type	Number of Series	Objective Function	Number of Iterations	At least one Time Period Different
Relative	Annual	(Stock)	737	0.00%	-	0.00%
	Monthly	Flow	284	0.00%	-	0.00%
		Stock	378	0.00%	-	0.00%
	Quarterly	Flow	56	0.00%	-	0.00%
Trend	Annual	(Stock)	737	2.99%	2.99%	36.49%
	Monthly	Flow	284	14.44%	13.03%	45.77%
		Stock	378	5.56%	4.49%	60.05%
	Quarterly	Flow	56	10.71%	12.50%	64.29%

However, the trend revision showed differences in the objective function, number of iterations, and in the revised series. The most intriguing pattern shows a large percentage of series that have an equal objective function value and number of iterations but the rounded series values were not equal. Further investigation found that the Fortran program outputs the ($j-1$) iteration solution instead of the j^{th} iteration solution as the final series. Yet, the objective function value and number of iterations listed in the ASCII output were for the j^{th} iteration. The SAS code uses the j^{th} iteration solution as the final series which does not follow the Fortran code.

Table 4: Percentage of Series that are Different for the Trend Revision by Objective Functions being Equal

				Percentage of Series Different		
Equal Functions	Frequency	Series Type	Number of Series	Objective Function	Number of Iterations	At least one Time Period Different
Yes	Annual	(Stock)	715	0.00%	0.13%	36.08%
	Monthly	Flow	243	0.00%	0.24%	41.97%
		Stock	357	0.00%	0.00%	57.98%
	Quarterly	Flow	50	0.00%	0.02%	60.00%
No	Annual	(Stock)	22	100.00%	95.45%	54.53%
	Monthly	Flow	41	100.00%	75.61%	70.73%
		Stock	21	100.00%	80.95%	85.71%
	Quarterly	Flow	6	100.00%	100.00%	83.32%

Table 4 presents the percentage of series different for the trend revision by objective functions being equal. The percentages show that for a large percentage of series, there is no difference between the last (j^{th}) and next to last ($j-1$) iterations in the rounded series. But, the programs still appear to take slightly different paths to obtain a solution for a small number of series. Yet for an even smaller number of series the different paths obtain the same rounded solution.

To evaluate the differences in the objective functions, a measure using the stopping rule ratio was calculated as:

$$rfunc = \frac{F_T(y_t^{(SAS)}, t = b_1, \dots, e_m)}{F_T(y_t^{(Fortran)}, t = b_1, \dots, e_m)}$$

If $rfunc$ were less than one, the SAS program would have achieved a smaller objective function. The differences in iterations were calculated as the Fortran value minus the SAS value. Therefore, a positive difference would mean SAS took less iterations. Table 5 shows the mean, median, minimum, and maximum of the ratio of objective functions and the differences in iterations. No particular program (Fortran or SAS) provides a “better” trend revision solution consistently. A few E-commerce series were problematic because the procedure stopped due to the number of iterations instead of the stopping rule. The E-commerce series is a newer indicator; therefore, the original series and benchmarks series are more volatile.

Table 5: Mean, Median, Minimum, and Maximum Difference between Fortran and SAS for Trend Revision

	Frequency	Series Type	Mean	Median	Minimum	Maximum
Objective Function	Annual	(Stock)	1.00017743	1.00000000	0.94784374	1.09516232
	Monthly	Flow	1.00004220	1.00000000	0.99929010	1.00745363
		Stock	1.00000368	1.00000000	0.99965685	1.00075982
	Quarterly	Flow	0.99998907	1.00000000	0.99890836	1.00066107
Iterations	Annual	(Stock)	0	0	-68	25
	Monthly	Flow	0	0	-9	33
		Stock	0	0	-27	30
	Quarterly	Flow	0	0	-16	23

In addition to the objective function, the revisions in the growth rates were calculated as:

$$ut_t^{Program} = \left(\frac{y_{t+1}^{Program}}{y_t^{Program}} - \frac{x_{t+1}}{x_t} \right) * 100 \text{ for } t = 1, \dots, n$$

and the difference in the absolute revisions calculated as:

$$DT_t = \left| ut_t^{Fortran} \right| - \left| ut_t^{SAS} \right| \text{ for } t = 1, \dots, n .$$

All side-by-side comparison box plots of the revisions showed very similar distributions for Fortran and SAS (see Figure 5). A time series plot of the differences in absolute revisions may show which program preserved the growth rates better for a particular series. For example, if the majority of the values (DT_t , for $t = 1, \dots, n$) were greater than zero SAS would have smaller revisions to the growth rates. All of these time series plots showed no strong preference for either software implementation, as is shown in Figure 6.

It appears that two local minimum solutions can be very similar in preserving the growth rates. Numerical precision error may play a part in the differences because of the sensitivity of steepest feasible descent. Di Fonzo and Marini (2011) provide evidence that the trend object function is non-convex, meaning that a local minimum is not necessarily the global minimum. Further research into the nature of the objective function and more robust non-linear optimization procedures may help reduce these types of differences.

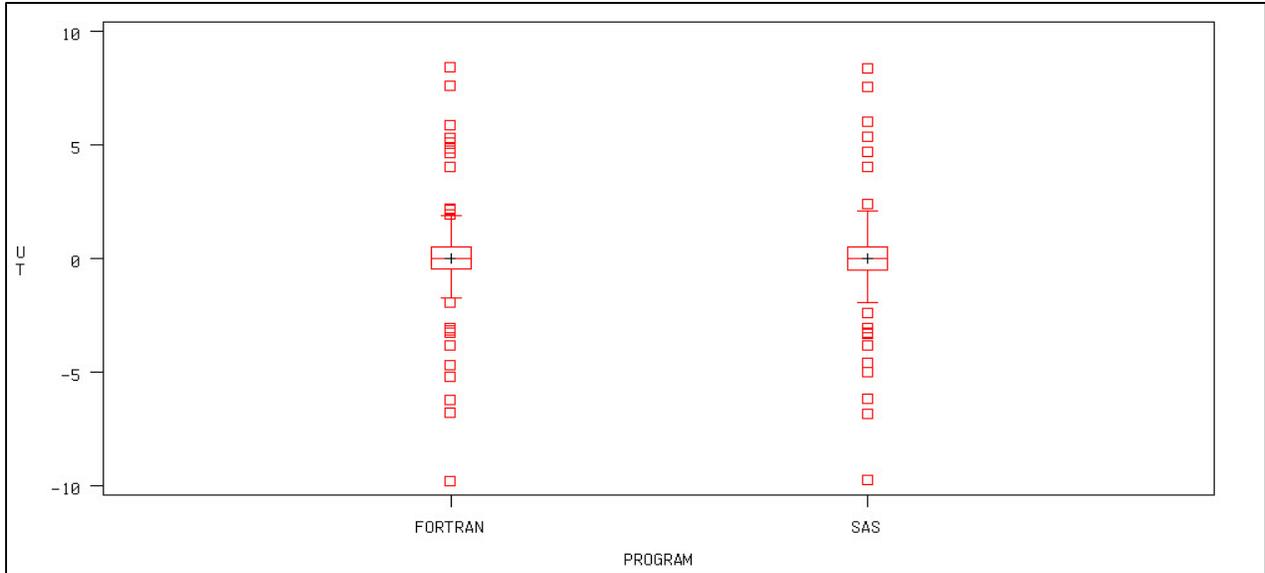


Figure 5: Comparison Box Plot of Revisions by Software Implementation

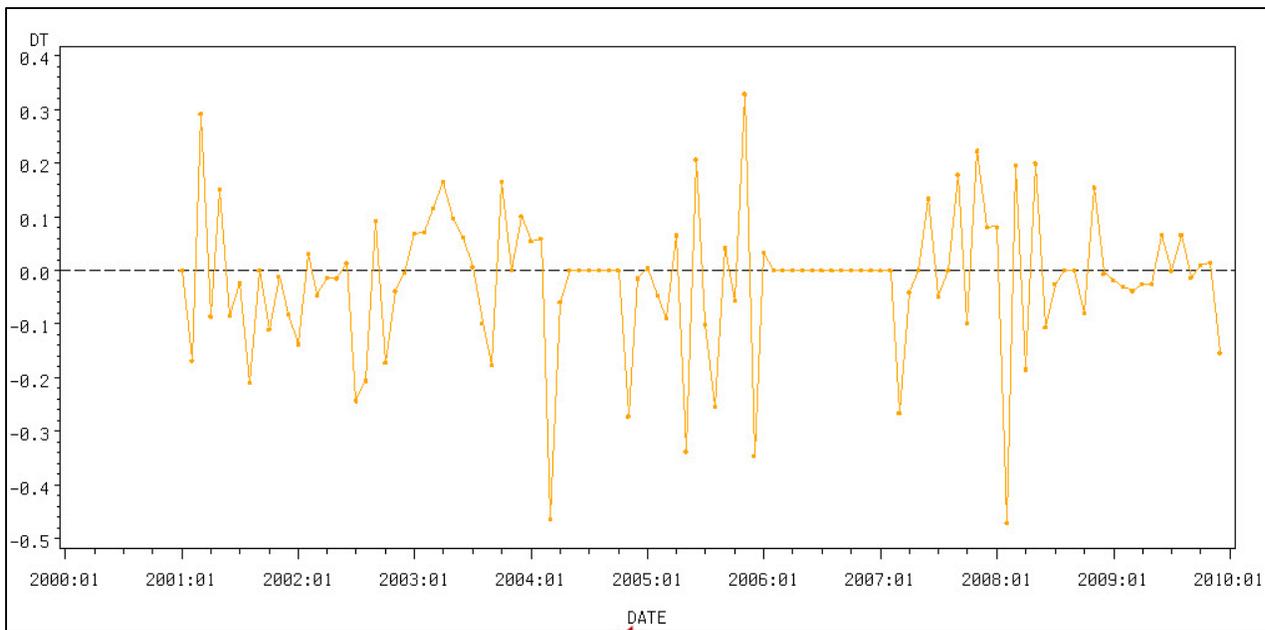


Figure 6: Time Series Plot of Difference in Absolute Revisions

Several measures were calculated to evaluate the magnitude of the differences in the series values. First, the absolute difference was calculated as

$$absdiff_t^i = |y_t^{i,Fortran} - y_t^{i,SAS}|,$$

where i represents a time series and t is the time period within series i . The percent absolute difference calculated as

$$abspct_t^i = \frac{|y_t^{i,Fortran} - y_t^{i,SAS}|}{y_t^{i,Fortran}},$$

where i and t represent the same as above.

The averages per series calculated as

$$avgdiff^i = \frac{\sum_{t=1}^{n_d^i} absdiff_t^i}{n_d^i} \text{ and } avgpct^i = \frac{\sum_{t=1}^{n_d^i} abspct_t^i}{n_d^i},$$

where n_d^i is the number of time periods within a series where the difference between the Fortran and SAS revised series is not zero. The maximum absolute difference and maximum percent absolute difference were also found per series as:

$$mxdiff^i = \max(absdiff_t^i, t = 1, \dots, n) \text{ and } mxpct^i = \max(abspct_t^i, t = 1, \dots, n).$$

Summary measures by groups of series were also calculated. The first grouping divided the series between series where the objective functions were equal and those that were not. The first group, consisting of those series that were equal, represents the differences between the last (j^{th}) and next to last ($(j-1)$) iterations in the rounded series. The second group, consisting of those series that were not equal, represents all other possible differences. Each of the two groups were also broken down by frequency and series type. The means of the average absolute difference and average percent absolute difference were calculated as

$$\overline{avgdiff} = \frac{\sum_{i=1}^Z avgdiff^i}{Z} \text{ and } \overline{avgpct} = \frac{\sum_{i=1}^Z avgpct^i}{Z}$$

where Z is the number of series within frequency, series type, and if objective functions are equal group. Also, the means of the maximum absolute difference and maximum percent absolute difference were calculated as

$$\overline{mxdiff} = \frac{\sum_{i=1}^Z mxdiff^i}{Z} \text{ and } \overline{mxpct} = \frac{\sum_{i=1}^Z mxpct^i}{Z}.$$

The maximums were also found for all four measures within each group.

Table 6 provides a summary of the average absolute and average percent absolute differences. The average magnitude of the differences appears to be small for all groups. The differences between one iteration at the end of the algorithm appear to be the smallest except for few annual series that have slightly larger differences. For those series where the objective functions are different, monthly stock series show the largest average absolute differences.

Table 6: Summary of the Average Absolute Difference and Average Percent Absolute Difference

Objective Functions Equal	Frequency	Series Type	Number of Series	Average Absolute Difference		Average Percent Absolute Difference	
				Mean	Max	Mean	Max
Yes	Annual	(Stock)	258	1.37	10.75	0.04%	4.34%
	Monthly	Flow	102	1.01	1.20	0.04%	0.21%
		Stock	207	1.00	1.26	0.03%	0.34%
	Quarterly	Flow	30	1.03	1.31	0.01%	0.05%
No	Annual	(Stock)	12	1.42	3.00	0.32%	1.43%
	Monthly	Flow	29	1.13	1.93	0.52%	8.71%
		Stock	18	2.57	12.31	0.12%	0.74%
	Quarterly	Flow	5	1.43	2.00	0.53%	2.31%

Table 7 shows the summary of the maximum absolute and maximum percent absolute differences. Once again, the differences are mostly small. For example, all series that had a maximum percent absolute difference greater than one percent always had a maximum absolute difference of one. Suggesting that the series with higher percent differences were of small magnitude (i.e. $1/7 = 14.29\%$). For all the series with a maximum absolute difference greater than two, all maximum percent absolute differences were less than one percent. Once again, the larger

maximum absolute differences seem to occur for the annual and monthly stock series. Yet, the larger percent absolute differences generally occur in small magnitude monthly flow series.

Table 7: Summary of the Maximum Absolute Difference and Maximum Percent Absolute Difference

Objective Functions Equal	Frequency	Series Type	Number of Series	Maximum Absolute Difference		Maximum Percent Absolute Difference	
				Mean	Max	Mean	Max
Yes	Annual	(Stock)	258	1.68	17.00	0.04%	4.35%
	Monthly	Flow	102	1.05	2.00	0.04%	0.24%
		Stock	207	1.02	2.00	0.03%	0.34%
	Quarterly	Flow	30	1.20	2.00	0.01%	0.05%
No	Annual	(Stock)	12	1.75	4.00	0.35%	1.43%
	Monthly	Flow	29	2.07	11.00	0.77%	14.29%
		Stock	18	6.33	33.00	0.19%	0.99%
	Quarterly	Flow	5	1.80	3.00	0.89%	4.00%

These larger absolute differences generally clustered within certain years or benchmark periods within a series. A plot of the absolute differences from a monthly stock series illustrates this pattern (see Figure 7). Upon reviewing the revised to original ratio plots, large differences in the series seemed to occur when large changes in the ratios happen. Figure 8 shows the revised to original ratio plot for the same monthly stock series in Figure 7. A large change occurs in the ratio from December 2008 to December 2009, which is the same time span for the large absolute differences in the series values (see Figure 7).

This pattern showed up consistently in all the series that had larger absolute differences. Even for flow series and the annual series that were different only due to one iteration at the end. Although, the larger differences generally occurred in stock and annual series suggesting their less binding nature may explain the differences.

4. Conclusion

Economic Programs at the U.S. Census Bureau use the Causey-Trager procedure for benchmarking monthly and quarterly series to annual series and to the Economic Census every five years. This procedure, developed thirty years ago, uses an iterative, nonlinear technique known as steepest (gradient) feasible descent to obtain the benchmarked series by optimizing a measure of growth-rate preservation. Even though the software implementation of the procedure has changed a little with each new programming language, the methodology has not changed. The latest language being user defined codes in SAS.

Several issues arose when writing and testing the new code. The most problematic being inconsistencies between the actual Fortran code and documentation. One issue dealt with the actual time span that is benchmarked versus the carry forward/backward factors. Another issue involved the Fortran code providing the next to last iteration instead of the last iteration as the solution. The differences between values in the series due to one iteration at the end were small with approximately 50 percent of the series being the same when rounded and the rest being less than one percent different (except for a small number of annual series). Although, based on inconsistencies and larger differences between the $j-1$ iteration (ASCII Fortran) and the j iteration (SAS) with regard to stock and annual series, research has begun into a method described by Quenneville (2010) using natural cubic splines.

The last issue questioned the robustness of the method. Thirty years ago, the method of steepest feasible descent was considered one of the best options based on memory constraints and the speed of computers. However, today, non-linear practitioners describe the method as inefficient and non-robust. Other non-linear procedures have been researched and compared to the current methodology and ASCII Fortran software implementation (see Brown (2010) and Di Fonzo and Marini (2011)). Generally, the Causey-Trager method showed comparable results except for series with large differences from the benchmarks. When compared to the multiplicative Cholette-Dagum regression-based benchmarking method, the Causey-Trager method preserves the growth rates slightly better (Titova (2010)). Research continues to find the most robust benchmarking method that meets the benchmarking goals of the Census Bureau along with better diagnostics in finding the best method.

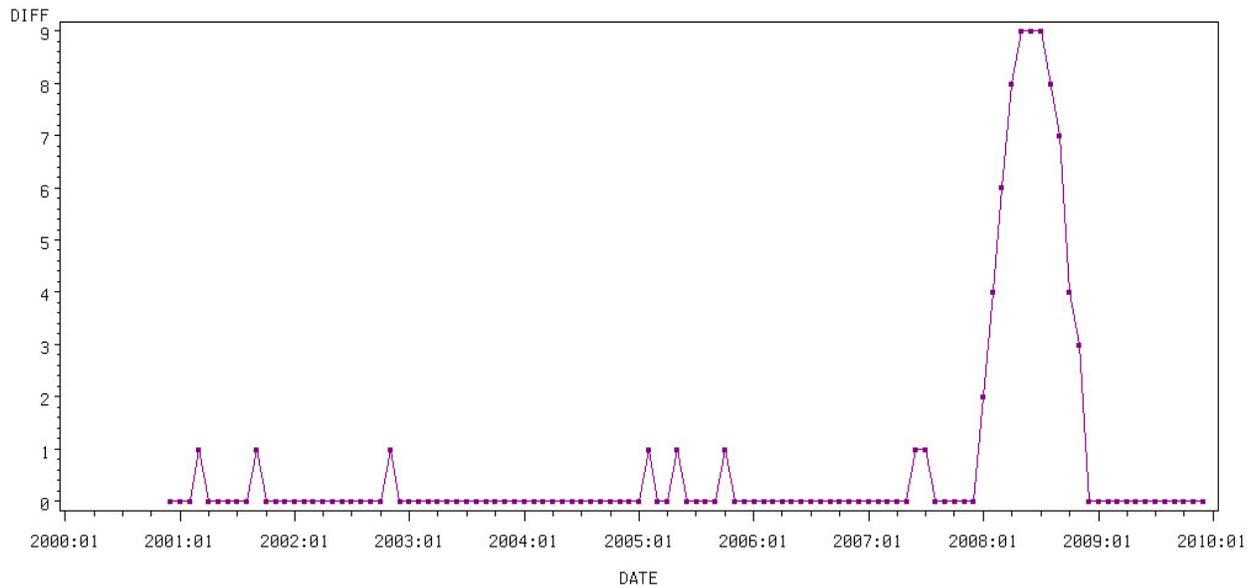


Figure 7: Time Series Plot of Absolute Differences for a Monthly Stock Series

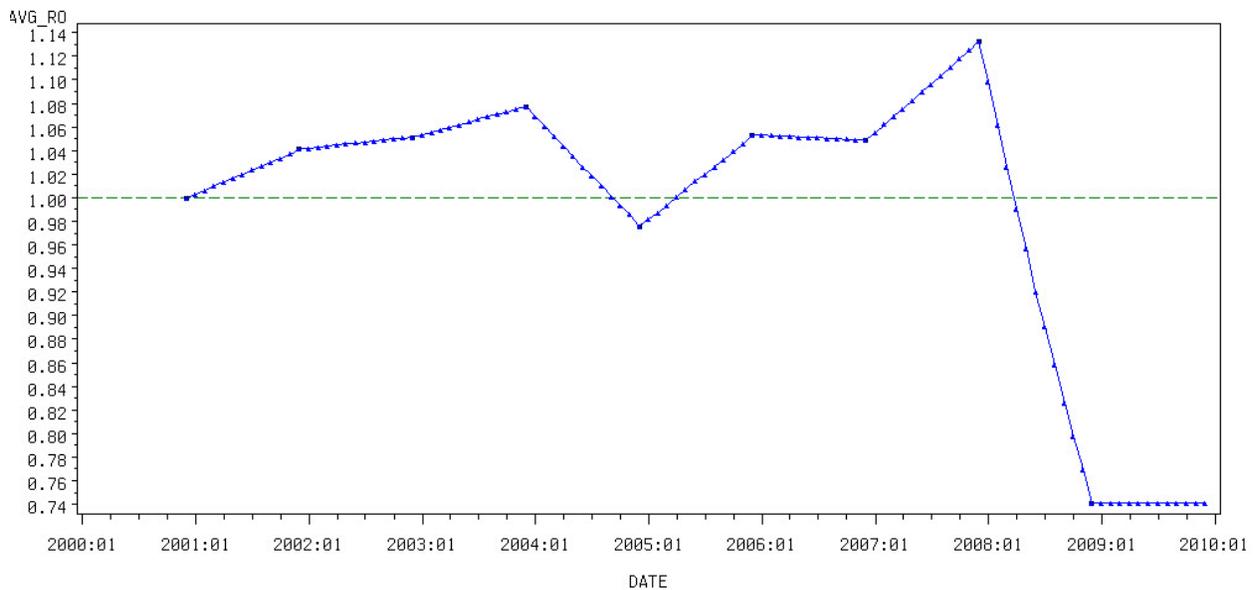


Figure 8: Plot of Revised to Original Ratios for a Monthly Stock Series

Acknowledgments

The author thanks Xijian Liu and David Kinyon for their careful review of earlier versions of this paper. The author also thanks Brian Monsell for a better understanding of the Fortran code and the staff from the Manufacturing and Construction Division (MCD) and the Service Sector Statistics Division (SSSD) for providing help in understanding their time series.

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. © indicates USA registration.

References

- Bozik, J.E. and Otto, M.C. (1998), Benchmarking: Evaluating Methods that Preserve Month-to-Month Changes. Statistical Research Report No. 88/07, U.S. Census Bureau. Available online at <http://www.census.gov/srd/papers/pdf/rr88-07.pdf>.
- Brown, I. (2010), An Empirical Comparison of Constrained Optimization Methods for Benchmarking Economic Time Series. In *JSM proceedings*, Business and Economic Statistics Section. Alexandria, VA: American Statistical Association, pp. 2131-2143.
- Causey, B. and Trager, M.L (1981), "Derivation of Solution to the Benchmarking Problem: Trend Revision." Unpublished research notes. U.S. Census Bureau. Available as an appendix in Bozik and Otto (1988).
- Cholette, P. (1984), "Adjusting sub-annual series to yearly benchmarks", *Survey Methodology*, **10**, 35-49.
- Dagum, E.B. and Cholette, P.A. (2006), *Benchmarking, Temporal Distribution, and Reconciliation Methods for Time Series*, New York: Springer, Lecture Notes in Statistics 186.
- Denton, F.T. (1971), "Adjustment of Monthly or Quarterly Series to Annual totals: An Approach Based on Quadratic Minimization." *Journal of the American Statistical Association*, **66**, 99-102.
- Di Fonzo, T., and Marini, M. (2011) *A Newton's Method for Benchmarking Time Series according to a Growth Rates Preservation Principle*. Department of Statistical Sciences, University of Padua, Working Paper Series, N. 7. <http://www.stat.unipd.it/ricerca/fulltext?wp=432>
- Quenneville B., Picard, F., Fortier, S., Interpolation Benchmarking and Temporal Distribution with Natural Splines. In *JSM proceedings*, Business and Economic Statistics Section, Alexandria, VA: American Statistical Association, pp. 106-120.
- SAS Institute Inc. 2008. **SAS OnlineDoc® 9.1.3**. Cary, NC: SAS Institute Inc.
- Titova, N., Findley, D., and Monsell, B.C (2010), Comparing the Causey-Trager Method to the Multiplicative Cholette-Dagum Regression-based Method of Benchmarking Sub-annual Data to Annual Benchmarks. In *JSM Proceedings*, Business and Economic Statistics Section, Alexandria, VA: American Statistical Association, pp. 3007-3021.
- Trager, M.L. (1982), "Derivation of Solution to the Benchmarking Problem: Relative Revision." Unpublished research notes, U.S. Census Bureau. Available as an appendix in Bozik and Otto (1998).