INERTIAL BALANCE DATA ANALYSIS

(Rev 2)

MODEL

We will assume that the device functions like a simple spring-mass system, the period of oscillation T of which will then be given by

$$T = \frac{2\pi}{\sqrt{k}}\sqrt{M+m} \tag{1}$$

where k is the Hooke spring constant, m is the mass of the "tray" that holds the masses, and M is the added, usually unknown, mass. The tray mass m could also be a known "dummy" mass, added to slow down the oscillation to make it easier to measure. As will be seen below, this added mass will have the additional benefit of linearizing the T vs M curve. Note that in this formulation, if the added mass M is zero, the device will still have a finite oscillation period. The reason for separating out the k factor will be seen later (it usually appears inside the square root with the masses). The tray mass m will turn out to be an important factor; it is not usually defined separately, and apparently it can vary significantly from one device design to another.

The device will of course have some damping, since, once set in motion, it will not oscillate forever. It has been noted that "in most practical cases the damping has but a slight influence on the frequency of free vibration" (*Fundamentals of Vibration Analysis*, N. O. Myklestad, Dover, 2018, pp.74-75). Any damping that occurs in the oscillation of the device will be implicitly included in the calibration data, assuming that the procedure for calibration is the same as that for measuring an unknown mass.

CALIBRATION

This is the process of establishing the relation between the input to a device and its corresponding output. We do this through the use of a set of known inputs (here, masses) for which we observe the instrument's response (here, period of oscillation). From this data we need to find an "interpolating function," which is a mathematical expression that allows us to estimate values of the device output in between the several settings used in the calibration.

This interpolating function is usually found with a "regression analysis." The latter is often called "finding the line of best fit" or something similar. Actually the objective is to estimate the parameters in a mathematical model so that, in a certain sense, the model will then "best" represent the observed data. This is done by adjusting the parameter(s) in the model in such a way that the sum of the squared differences between the data and the function is minimized. Hence the term "least squares."

The function used for interpolation can be any arbitrary expression, like a low-order polynomial, or it might be based on a "physics model" that represents the behavior of the device. In the latter case, when we estimate the parameters in that model they will usually have some physical interpretation, but this is a bonus, and not the principal purpose of the calibration.

A major requirement for a calibration interpolation function is that it must be readily "invertible." This is because, in using the device, we will observe its *output*, from which we would like to estimate the *input* that caused it. The interpolation function was done the other way, using defined *inputs* to generate the corresponding observed *output.* Some otherwise excellent interpolating functions, that represent the calibration data very well, are not practical for use, due to difficulty in inverting them.

In the case of the inertial balance, we use known masses M, and observe the corresponding periods T. (We might also, as will be seen below, need to estimate the mass of the "tray" m.) In using the balance, we would observe the period T of an unknown mass M, and our calibration equation might have been something of the form $T = a + bM$, for example. Then, having observed the period T for the unknown mass, we would invert that calibration equation to obtain an estimate of the mass from $M = (T - a) / b$. Listed below are several approaches for this analysis.

UNCERTAINTY ASPECTS

When we have a function of random variables, as we will when using the inverted calibration function, we need to apply the "propagation of error" (POE) formula; see the Wikipedia article "Experimental Uncertainty Analysis" for a derivation:

$$z = f(x_1, x_2, x_3, ...) \qquad \mathbf{Var}(z) \approx \sum_{i,j} \left(\frac{\partial z}{\partial x_i} \right) \left(\frac{\partial z}{\partial x_j} \right) \mathbf{Cov}(x_i, x_j) \tag{2}$$

Here the summation is taken over *all combinations* of the variables *x*, and the covariance of any *x* with itself is its variance. The "standard deviation" (σ) is the square root of the variance, and is the value we quote as the "uncertainty" in our result, like this: $z \pm \sigma$. Note that "error" is an unfortunate choice of words, implying some sort of mistake, but this terminology is commonly used. Perhaps a better phrase would be the "propagation of uncertainty."

The idea of POE is that, given a random variable, which has inherent statistical properties, if we do some form of mathematical operation(s) on it, we will be changing those properties. (This is why it is best, if possible, to avoid performing math operations, or transformations, on observed data.) Eq(2) is an approximation for one of those properties, its variance. It is good practice to check the POE calculation, which can become quite cumbersome, with a so-called "Monte Carlo" simulation. This has been done to verify that Eq(2), in each case below, provides a good estimate of the respective uncertainty for that case.

Here is an example of a Monte Carlo calculation of the estimated mass (*mhat*), for Method 1, defined in the next section:

```
mhat =  ( normrnd(muT,sigT,[npts,1]) - normrnd(beta0,beta0sig,[npts,1]) ) ...
                    ./ normrnd(beta1,beta1sig,[npts,1]);
```

We take *npts*=20000 random samples from separate Normal distributions for each of the period *T, beta0,* and *beta1* and operate on them as defined in Method 1. The mean values *beta0* and *beta1* come from the calibration, along with their uncertainties *beta0sig* and *beta1sig*. The mean and standard deviation of the resulting 20000 *mhat* values are then found, and compared to the known mass (which was used to calculate the mean value for *T, muT*) and the standard deviation (σ) from the POE, Eq(2). This process has verified the POE calculations for all methods discussed below; with such a large sample size the observed and POE sigmas agree well. The nature of this comparison is such that even a slight error in the POE would produce a noticeably incorrect sigma. Also, it is interesting to note how in years past this large a simulation sample size would have required a very long execution time, while the studies done here typically took only about 0.05 seconds each, on a rather ordinary machine.

Another output of the Monte Carlo run is a histogram of the estimated masses. This gives an indication of the probability distribution function (PDF) of the estimates. It is common practice to *assume* that these estimates follow a Normal distribution, but that is not necessarily the case; highly skewed PDFs are possible. It happens in the cases studied here that the PDFs do appear to follow a Normal distribution, in all cases (see Fig.1 for an example). This is a consequence, apparently, of the fact that, due to the large sample size used in calibration, the variances of the parameters used in the mass-estimation calculations are quite small. It seems that when the "coefficient of variation" (aka relative error) of the parameters in a POE calculation are small, the PDF of the estimated quantity will be approximately Normal (this needs further research).

Testing with larger sigmas, such as those that would result from more realistic sample sizes, did produce non-Normal, slightly skewed PDFs. For example, the "Burr" distribution worked well in some cases, and the Gamma distribution in others. The reason to be interested in the PDF shape is for calculation of a "confidence interval" (CI) on the estimated mass. The departure from Normality is not extreme, and Monte Carlo results indicate that the usual multiplier of 1.96 (2 is close enough for most work) will still produce the desired 95% coverage. If CI calculations are not to be done then this aspect (the PDF) can be ignored and the estimated mass of the unknown can be quoted simply as the point estimate plus or minus the POE sigma. For an approximate 95% CI we would quote the point estimate plus or minus *twice* the sigma, assuming Normality, which, again, the Monte Carlo results show is reasonable here, even with modest sample sizes.

CALIBRATION / ESTIMATION METHODS

The data used for calibration of the methods below is synthetic data generated using Eq(1), with parameters *k* and *m* based on measurements made with an actual inertial balance. That device has a large "tray" mass *m*, about 0.7 kg, and a *k* value about 180 N/m. The mass added is over a range of 0.05 to 0.75 kg; the range of mass is relatively small, since it has been observed that larger masses can cause the device springs (thin metal strips) to deform and cease to act as simple springs.

The Normally-distributed "noise" added to the calculated periods has a standard deviation of 0.01 s. This was found to produce plausible variations, similar to those that would be expected in measuring *T*. On the other hand, the number of calibration mass levels (15) and the number of replicates at each level (50) are both much larger than would be the case in realistic experimentation; this was done so that the properties of the methods can be explored better. A more typical experiment might have 5 mass settings with 5 or perhaps 10 replicates at each level.

METHOD 1 — *Simple Linear*

Calibration model

$$T = \beta_0 + \beta_1 M$$

Estimation model

$$\hat{M} = \frac{T - \beta_0}{\beta_1}$$

Estimation method

```
xdesign = [ ones(nlevels,1) thex' ];
[ beta, betasig ] = lscov( xdesign, Tmean' );
```

The variable t*hex* is a vector containing the *nlevels* of calibration masses *M*. The variable *Tmean* is a vector of the means (averages) of the replicated *T* measurements at each mass *M*. The variable *beta* is a vector of the regression coefficients, and *betasig* holds their corresponding uncertainties. So here the *x*-variable is *M* and the *y*-variable is *T*. (The code in this document is written in Matlab/Octave; the latter is a free, open-source version of Matlab.)

Uncertainty (POE) calculation

```
mhatvarPOE = (1/beta1^2) * (beta0sig^2 + Tsig^2) + ...
              beta1sig^2 * (Tmu-beta0)^2 / beta1^4;
mhatsigPOE = sqrt( mhatvarPOE )
```

A "hat" over a symbol is notation for an estimate; *M-hat* is the estimated mass of the unknown. By convention in regression analysis, *beta0* is the intercept and *beta1* is the slope. These will be *beta(1)* and *beta(2)* respectively in the regression code fragment above. *mhatvarPOE* is the variance of the estimated mass, obtained via Eq(2). *Tmu, Tsig* are the mean and standard deviation of the *T* values, found using the replicated *T* measurements at each *M*. (The Greek letter "mu" is used to denote a mean.) Note that there is no covariance term in this POE, even though the beta coefficients in linear regression have a nonzero covariance. It was found by numerical experimentation that a covariance term was not necessary in order to obtain good agreement between the POE and observed sigmas.

Discussion

This is a simple linear regression (SLR) application. It is best applied when the "tray" mass, or a known, added "dummy" mass, *m* is large enough to shift the *T* vs *M* curve to the left, which has the effect of linearizing the curve over the mass range of interest (usually 0 to 1 or 2 kg). When *m* is smaller, the *T* vs *M* graph has considerable

curvature at the lower end of the mass range, and this linear model will not apply there. Another approach then would be to use this linear model, but only over a restricted range of masses.

To illustrate these ideas, see Figs. 2, 3. In Fig. 2 there are three curves, with increasing *m* from bottom to top. The bottom graph would represent a very low-mass tray (0.05 kg), and there is considerable curvature, while the top graph has a massive tray, or dummy mass, (1.5 kg) and is essentially linear. The straight lines shown are the tangent lines at the midpoint of the mass range (0.5 kg) to help visualize the curvature. The tangent line is given by

$$T_{\text{tangent}} = \frac{2\pi}{\sqrt{k}}\sqrt{h+m} + \frac{\pi}{\sqrt{k}}\frac{M-h}{\sqrt{h+m}}$$

where *h* is the tangent point, 0.5 kg here. Fig. 3 shows the fractional difference between that tangent line and the actual function; the tray mass *m* increases in this plot from top to bottom. Note that the difference is exactly zero at the point of tangency, *h*. For the actual inertial balance used for this study, the middle graph applies (*m*=0.74 kg), and the fractional difference is less than 4% across this mass range. (This difference would represent a systematic error-assuming something is linear when in fact it isn't.) We could also use Fig. 3 to define a range over which the lightest-tray device could be used, by accepting, for example, no more than a 5% fractional error, and thus restricting the mass range to be about 0.25 to 1 kg.

METHOD 2 — *Square of period T, linear*

Calibration model

$$T^2 = \frac{4\pi^2}{k}\left[M+m\right] \;\Rightarrow\; \frac{4\pi^2}{k}m + \frac{4\pi^2}{k}M \;\Rightarrow\; \beta_0 + \beta_1 M$$

Estimation model

$$\hat{M} = \frac{T^2 - \beta_0}{\beta_1}$$

Estimation method

In this case we square the observed *T*, which we would rather not do, because it alters the statistical properties of the measured data. If the observed *T* data has a standard deviation of σ, when we square it, the POE Eq(2) leads to

$$z = x^2 \;\Rightarrow\; \sigma_z^2 = \left(\frac{\partial z}{\partial x}\right)^2 \sigma_y^2 = 4x^2\sigma_y^2 \;\Rightarrow\; \sigma_z = 2x\sigma_y$$

This says that the standard deviation of T^2 is not constant, it varies with *T*. See Fig 4, which shows the standard deviation of the sampled *T* (triangles), which are fluctuating around the known value of 0.01, and those of the T^2 (circles). The line shown is the expected behavior, from the calculation above. However, this variation turns out to be numerically small, for the magnitude of the measurement "noise" used here, and can be neglected.

A significant nonconstant variance would introduce considerable complexity into the analysis, if it is to be done correctly, and illustrates why we would rather not do math operations, like squaring, on observed random-variable data. (By contrast, such operations on "known" quantities, like the inputs used in calibration, here, the masses *M*, cause no problems.) One such complexity is that we should use a "weighted" regression analysis. Essentially, the data points with the smallest variability get more weight in the regression; the usual format for linear regression analysis assumes a constant variance across all the data points. Below is code for a weighted SLR:

```
V = diag( Tsig.^2 );
xdesign = [ ones(nlevels,1) thex' ];
[ beta, betasig ] = lscov( xdesign, Tmean', V );
```

The variables have the same meanings as in Method 1, except that the *Tmean* values are for T^2. The *x*-variable is *M* and the y-variable is T^2. Note that the regression parameters *beta* have physical interpretations, in terms of the Hooke spring constant *k*, and the "tray" mass *m*. Although the change in variance happens to be relatively small, so that

weighting is not strictly necessary, it will not hurt anything to use it. That is, if the variances are actually constant, the estimation results with and without weighting are essentially identical.

Uncertainty (POE) calculation

```
mhatvarPOE = 4*Tmu^2/beta1^2 * Tsig^2 + 1/beta1^2 * beta0sig^2 + ...
                  (Tmu^2 - beta0)^2 / beta1^4 * beta1sig^2;
mhatsigPOE = sqrt( mhatvarPOE );
```

METHOD 3 — *Square root of total mass, linear*

Calibration model

$$T = \frac{2\pi}{\sqrt{k}}\sqrt{M+m} \quad \Rightarrow \quad \beta = \frac{2\pi}{\sqrt{k}}$$

Estimation model

$$\hat{M} = \frac{T^2}{\beta^2} - m$$

Estimation method

Here we will take the "tray" mass *m* to have been separately measured. For some inertial balance devices the "tray" portion can easily be removed, placed on a scale or balance, and replaced. With this measurement, for the purposes of estimation we can then consider its mass *m* to be "known," although it will still have some associated uncertainty. Or, this *m* might be a known "dummy" mass, used to slow down the oscillation to make it easier to measure (and to linearize the *T* vs. *M* curve).

This case is an example of a regression situation that happens often in physics, namely, a "zero-intercept regression" (ZIR). When the model being estimated does not need an intercept, as here, it is best, statistically, to estimate the slope by itself. This is because we will obtain a smaller uncertainty in the slope estimate than would be the case if the usual simple linear regression (SLR) was done, which includes the slope *and* intercept. The basic statistical idea is that we shouldn't include terms in a regression model that don't need to be there- more is definitely *not* better. A ZIR, when appropriate, will give a more precise estimate of the slope than would SLR.

```
          xdesign = [ thex' ];  % ---- ZIR
[ beta, betasig ] = lscov( xdesign, Tmean' );
```

Here the *x*-variable *thex* will be √(*M* + *m*) and the *y*-variable is the period average *Tmean*. Note that, as a side benefit, the single estimated parameter *beta* has a physical interpretation, in terms of the Hooke spring constant *k*.

This form of estimation can also be programmed into a TI calculator:

$$\hat{\beta} = \frac{\sum xy}{\sum x^2} \qquad s^2 = \frac{\sum y^2 - \frac{\left(\sum xy\right)^2}{\sum x^2}}{n-1} \qquad \sigma\left(\hat{\beta}\right) = \sqrt{\frac{s^2}{\sum x^2}}$$

If the mass *m* is a "dummy" mass which is considered known, then set *msig* to zero in the POE below:

Uncertainty (POE) calculation

```
mhatvarPOE = 4*Tmu^2/beta^4 * (Tsig^2 + (Tmu/beta)^2 * betasig^2 ) + msig^2;
mhatsigPOE = sqrt( mhatvarPOE );
```

METHOD 4 — *Square root of total mass, nonlinear*

Calibration model

$$T = \frac{2\pi}{\sqrt{k}}\sqrt{M+m} \;\Rightarrow\; T = a\sqrt{M+b}$$

Estimation model

$$\widehat{M} = \frac{T^2}{a^2} - b$$

Estimation method

This is a nonlinear model (in the statistical sense) and we must use a special procedure to estimate the parameters *a* and *b* in this model. These parameters have physical interpretations. One way to estimate the parameters is to use the free, open-source, statistical language "R":

```
regr <- nls( they ~ a * sqrt(thex + b ), start=list( a=0.1, b=0.5 ), thedata )
```

Here *thex* are the masses *M*, and *they* are the values of *T* at each *M*. This code uses all the data points collectively, rather than the means (averages). The variable *thedata* is a construct in R that tells it where the data points are. The object *regr* implicitly contains the parameter estimates, their uncertainties, and other regression diagnostic information. These things are extracted as needed via the *$* operator.

Uncertainty (POE) calculation

This calculation is of exactly the same form, statistically, as in Method 3, so that POE also applies here, with the appropriate substitutions.

```
mhatvarPOE = 4*Tmu^2/a^4 * (Tsig^2 + (Tmu/a)^2 * asig^2 ) + bsig^2;
mhatsigPOE = sqrt( mhatvarPOE );
```

The nonlinear estimation process will return the parameter uncertainties *asig* and *bsig*.

METHOD 5 — *Square root of added mass, linear*

Calibration model

$$T = \beta_0 + \beta_1\sqrt{M}$$

Estimation model

$$\widehat{M} = \frac{\left(T-\beta_0\right)^2}{\beta_1}$$

Estimation method

This is a simple linear regression (SLR), with the *x*-variable being √(*M*), and the *y*-variable is *T*. This transformation of the known masses *M* does not cause any statistical difficulties, but happens that this model does not represent the data very well (see Figs. 13, 14), so it will not be pursued further.

METHOD 6 — *Mass raised to a power, nonlinear*

Calibration model

$$T = a + b M^c$$

Estimation model

$$\widehat{M} = \left(\frac{T - a}{b} \right)^{1/c}$$

Estimation method

This is another statistically nonlinear model, so we would again use R for the estimation. It turns out, however, that the coefficient *c* in this model is close to 1, so that we have essentially just a straight line. This model performs no better than Method 1, and is more difficult to estimate and to invert. So, it has not been developed further.

METHOD 7 — *Quadratic interpolation, linear*

Calibration model

$$T = a M^2 + b M + c$$

Estimation model

$$\widehat{M} = \frac{-b + \sqrt{b^2 - 4 a \gamma}}{2 a} \qquad \gamma = c - T$$

Estimation method

This is an example of a pure interpolation function, which has no connection to a physical model. It involves math operations only on the *known* variable, not the *observed* variable, so there are no statistical complexities. The coefficients have no physical interpretations, although the intercept *c* will be an estimate of the unloaded device period (Eq(1) with $M = 0$). The estimation model should look familiar. This model represents the data quite well, although, since the inversion is more complicated than the other methods, the POE for it is also rather complicated.

```
                xdesign = [ ones(nlevels,1) thex' thex2'];
        [ beta, betasig ] = lscov( xdesign, Tmean' );
```

In this code arrangement, which follows standard regression analysis formatting, we will have that

beta(1) = c (constant or intercept term); *beta(2) = b* (linear term); *beta(3) = a* (quadratic term)

and *thex2* is a vector containing the squares of the known masses M.

Uncertainty (POE) calculation

```
      c2  = c - muT; sigc2 = sqrt( sigc^2 + sigT^2 );
      gam = b^2 - 4*a*c2;

mhatvarPOE = ( ( b - sqrt(gam) )/( 2*a^2 )                  - ...
               c2/(a*sqrt(gam)) )^2 * siga^2                + ...
               1/(4*a^2) * ( b/sqrt(gam) - 1 )^2 * sigb^2 + ...
               1/gam * sigc2^2;
mhatsigPOE = sqrt( mhatvarPOE );
```

FIGURES

Figures 5-16 show the regression function with the data, and the corresponding residuals, for Methods 1-5,7 (6 is not used, it is little different from 1). The "error bars" on the data points show the standard deviation (sigma) of the random *T* values. The points plotted are the means of the *T* for each calibration mass. Only Methods 1 and 5 show a lack of fit (LOF) in their respective residuals (Figs. 6 and 14). The LOF for Method 1 is relatively minor (small magnitude residuals), but it is clear that the data is not completely linear. Method 5 has a distinct LOF that is obvious even without the residual plot.

Figure A below is a plot of the relative errors (RE) in the estimated mass as a function of that mass, for several calibration methods. All RE here are less than about 10%, and decrease to about 2% at higher masses. The reason for the upward trend in Method 7 (quadratic) is not evident at this writing, and needs further research. Note that these results used large calibration datasets (15 levels, 50 replicates) so that the uncertainties in the parameters are smaller than would be the case for more realistic sample sizes. This leads to smaller RE. More modest sample sizes will make these RE larger (move the graphs vertically), but they will keep the same general shape.

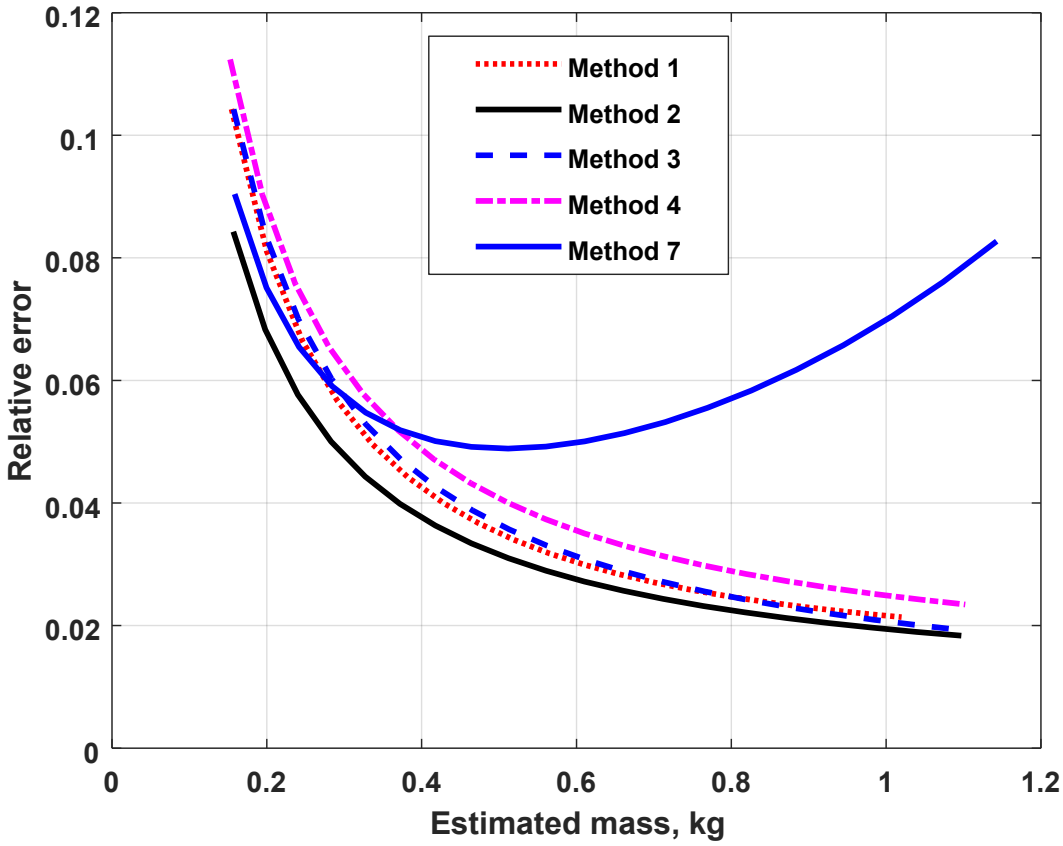

FIGURE A

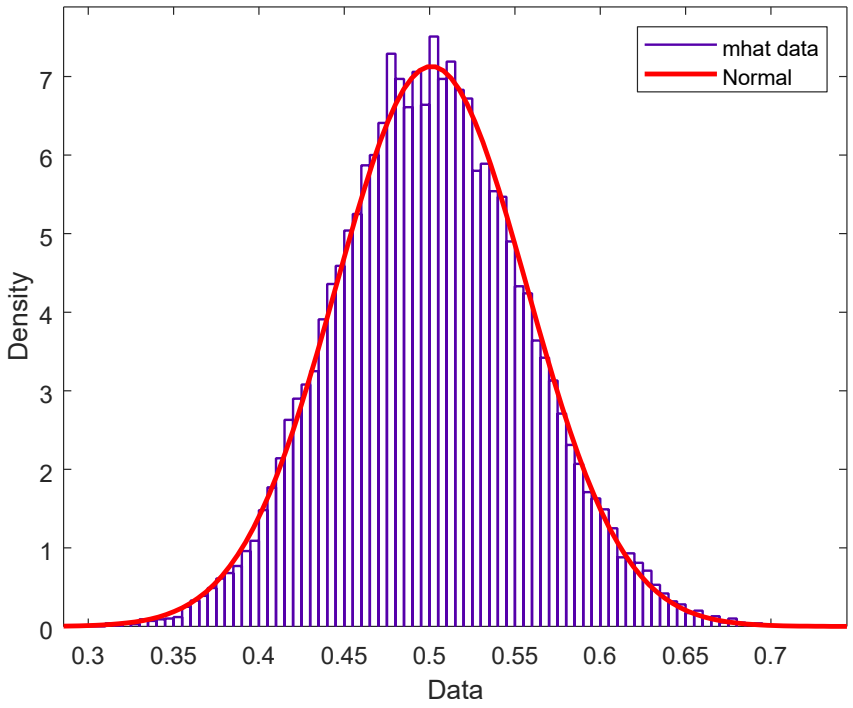

FIGURE 1

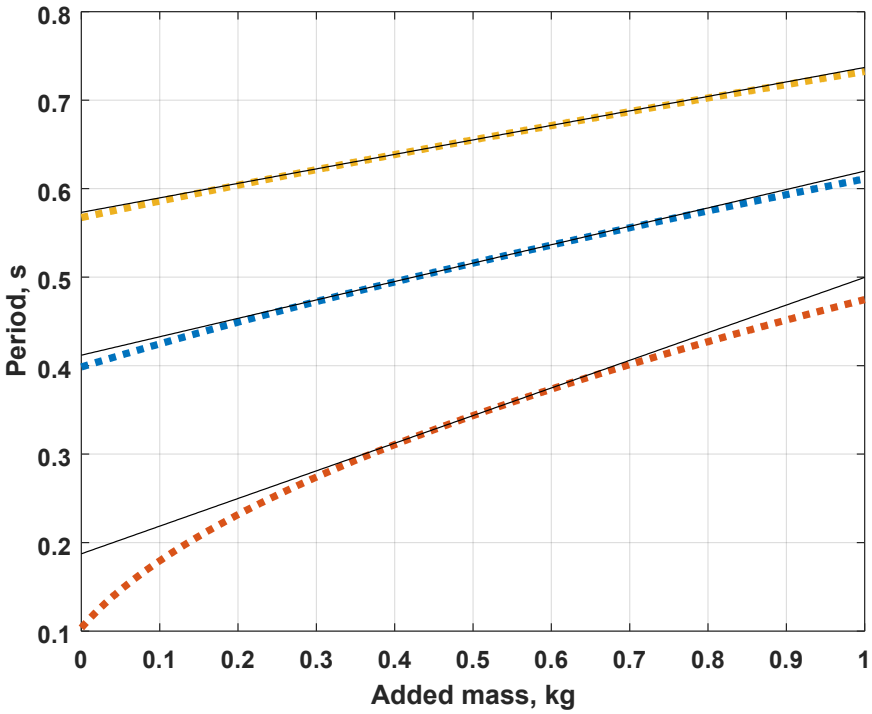

FIGURE 2

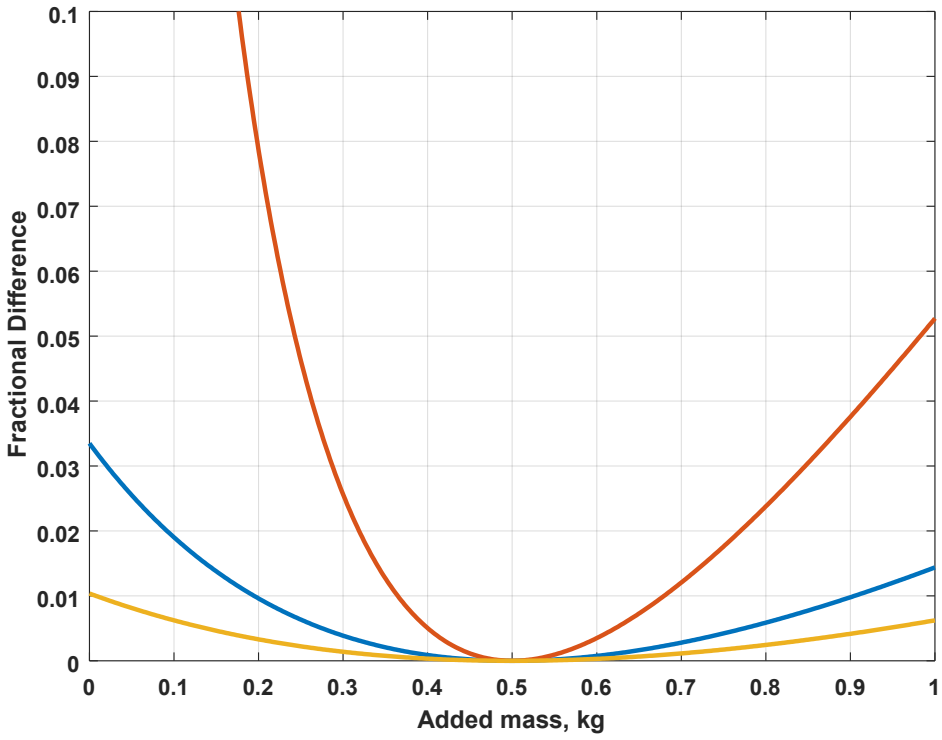

FIGURE 3

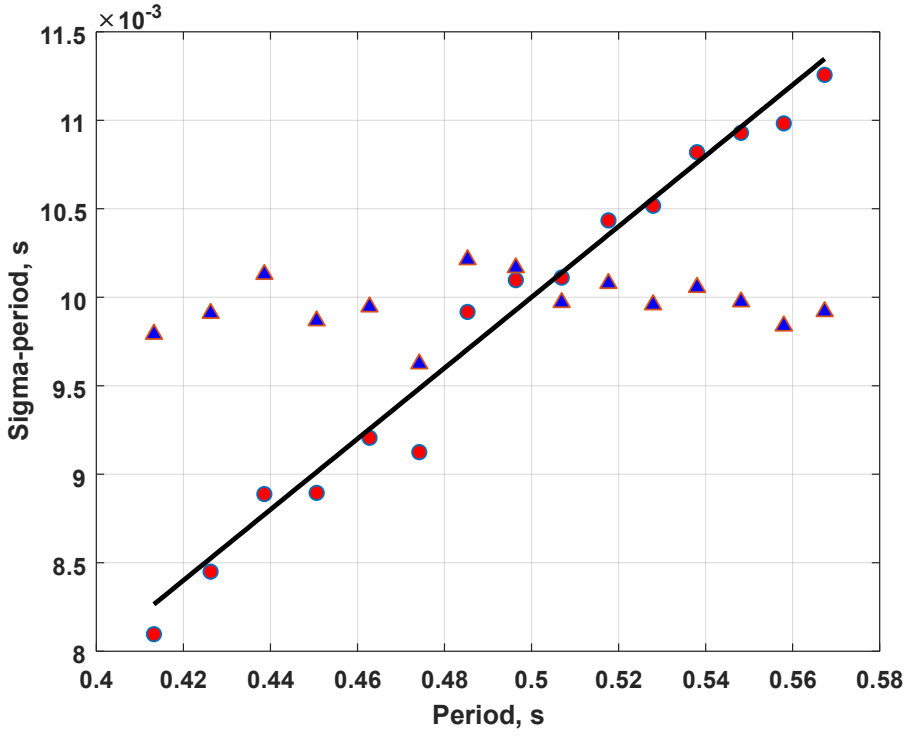

FIGURE 4

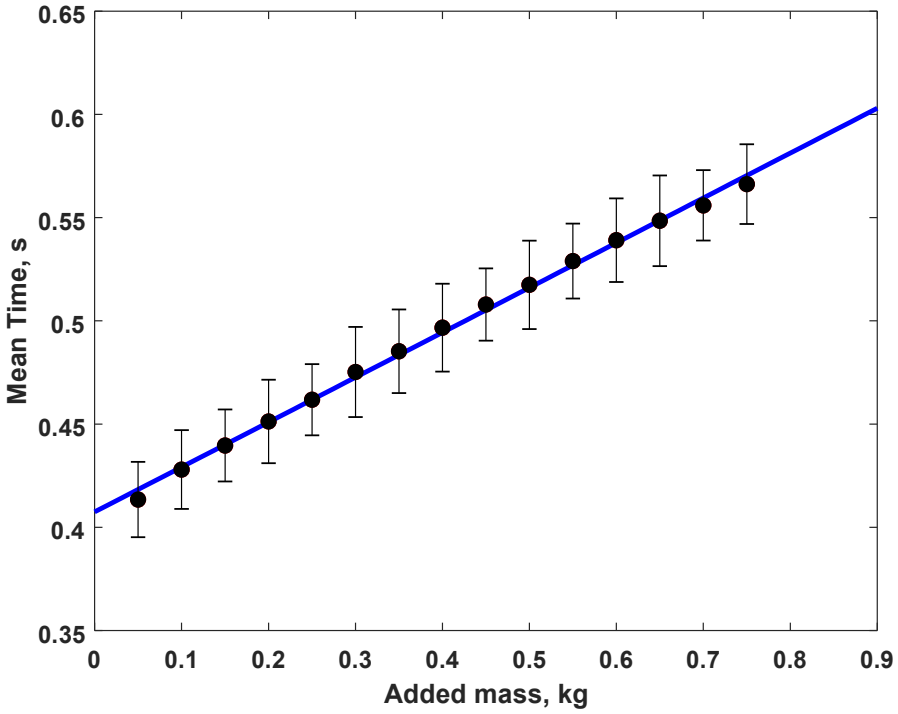

FIGURE 5 (Method 1)

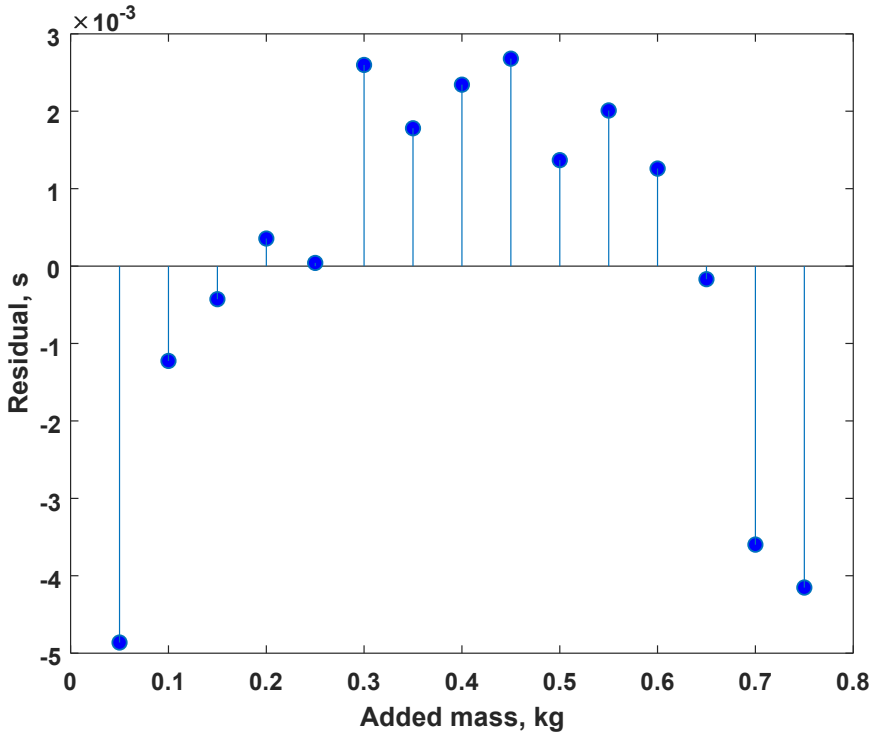

FIGURE 6

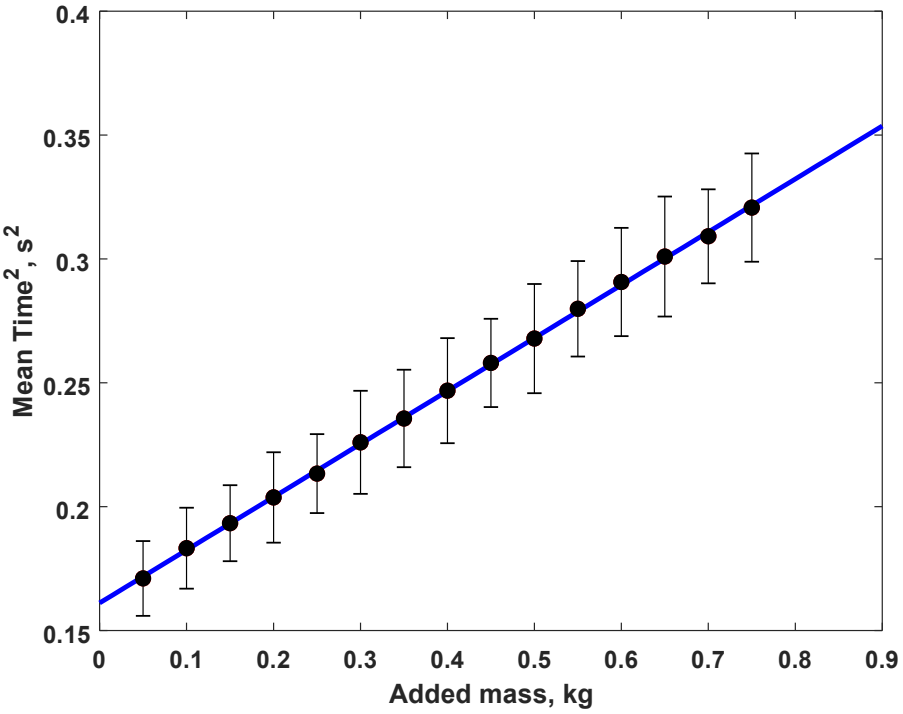

FIGURE 7 (Method 2)

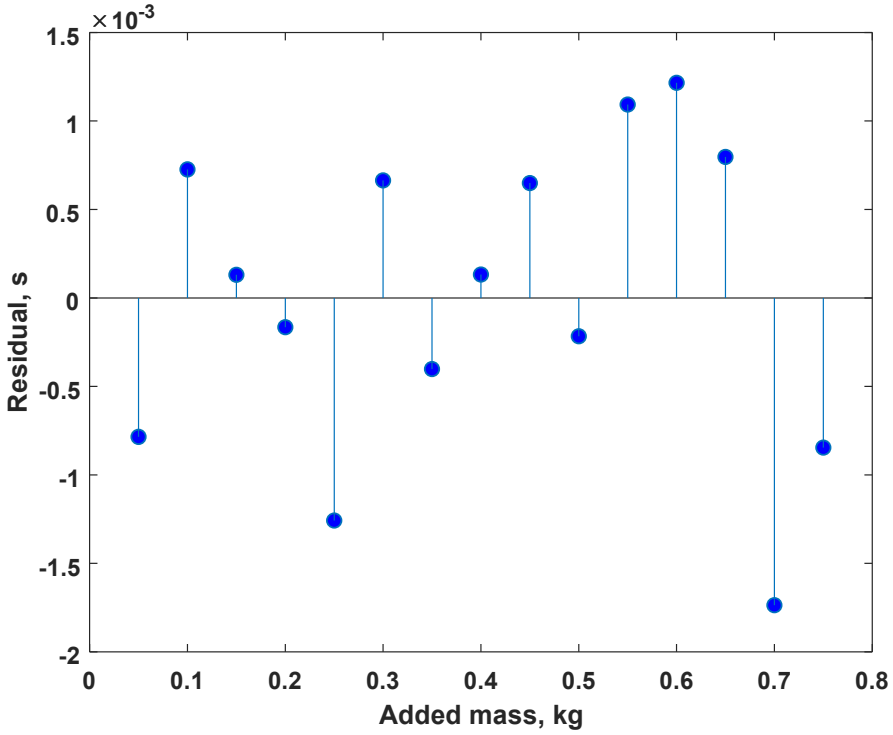

FIGURE 8

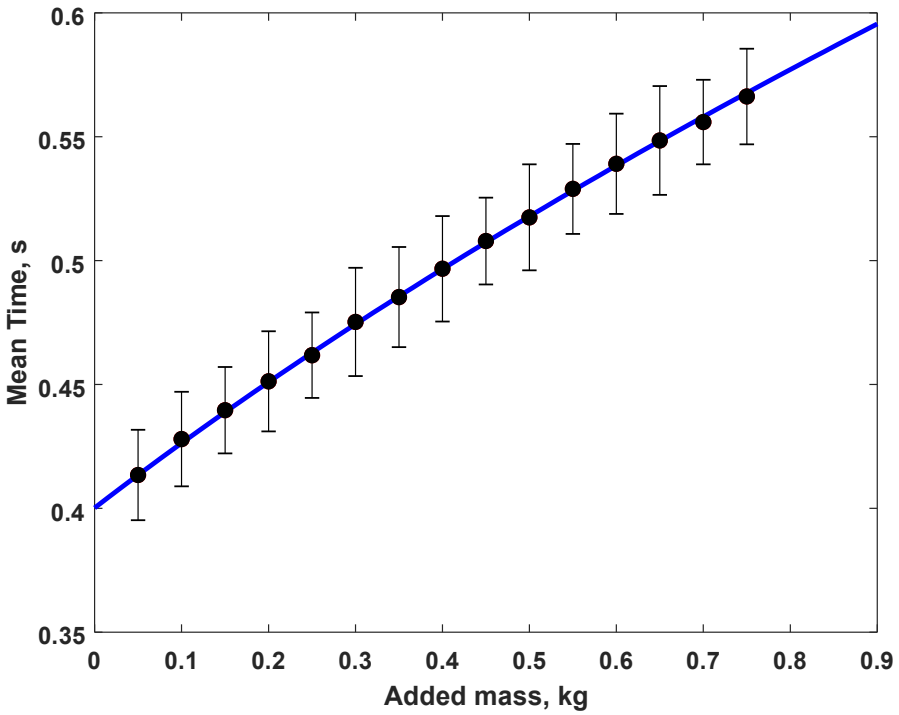

FIGURE 9 (Method 3)

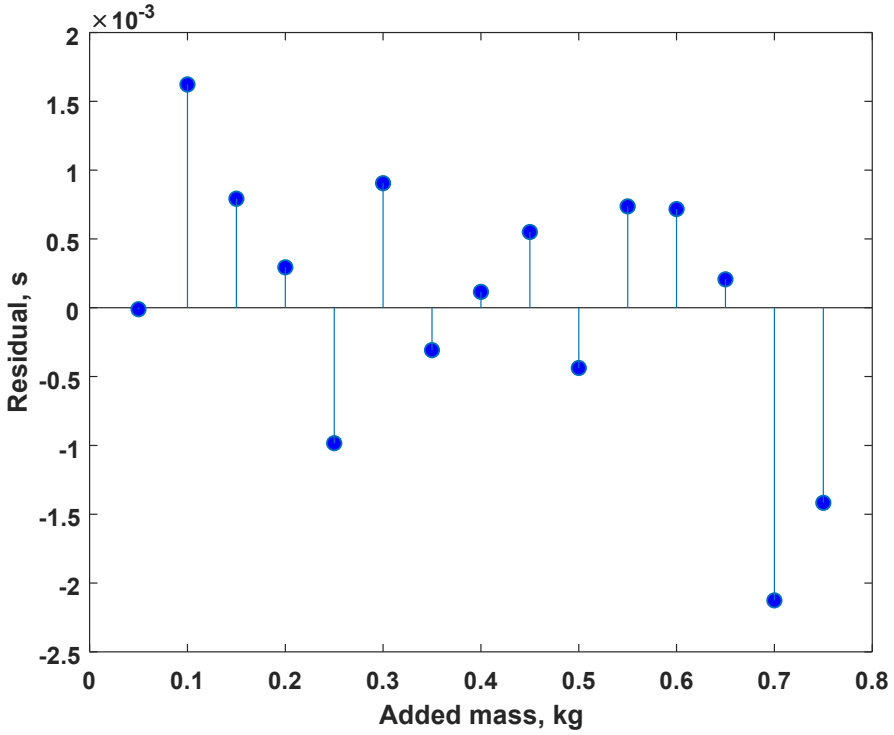

FIGURE 10

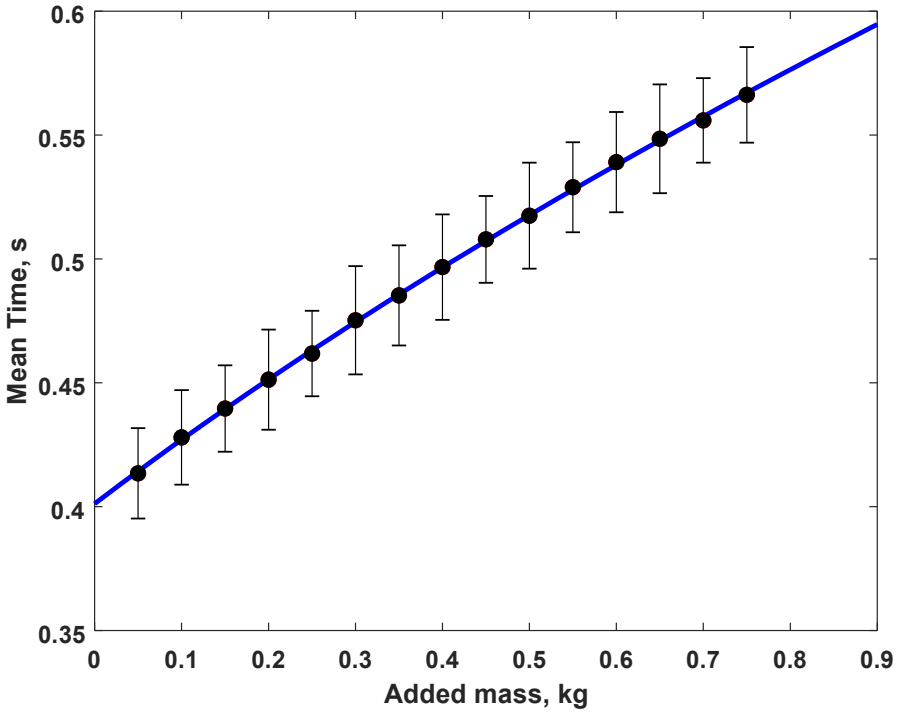

FIGURE 11 (Method 4)

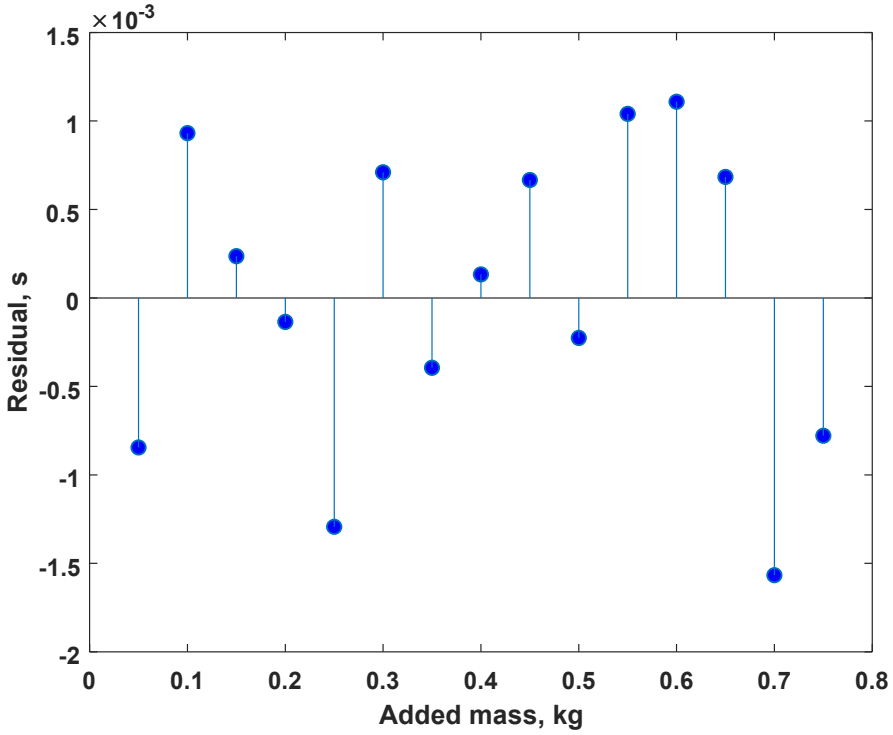

FIGURE 12

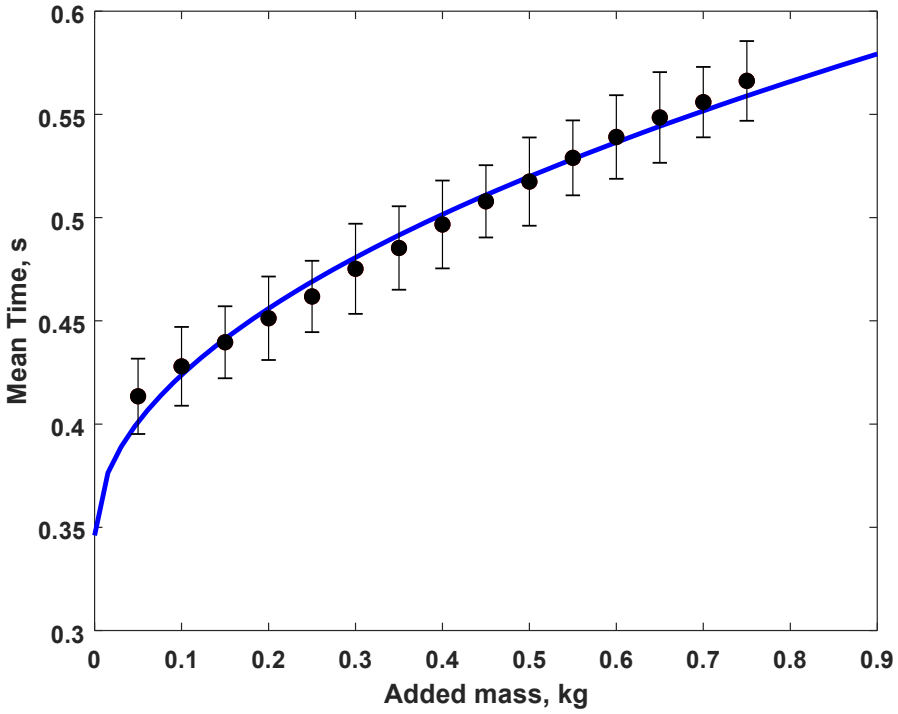

FIGURE 13 (Method 5)

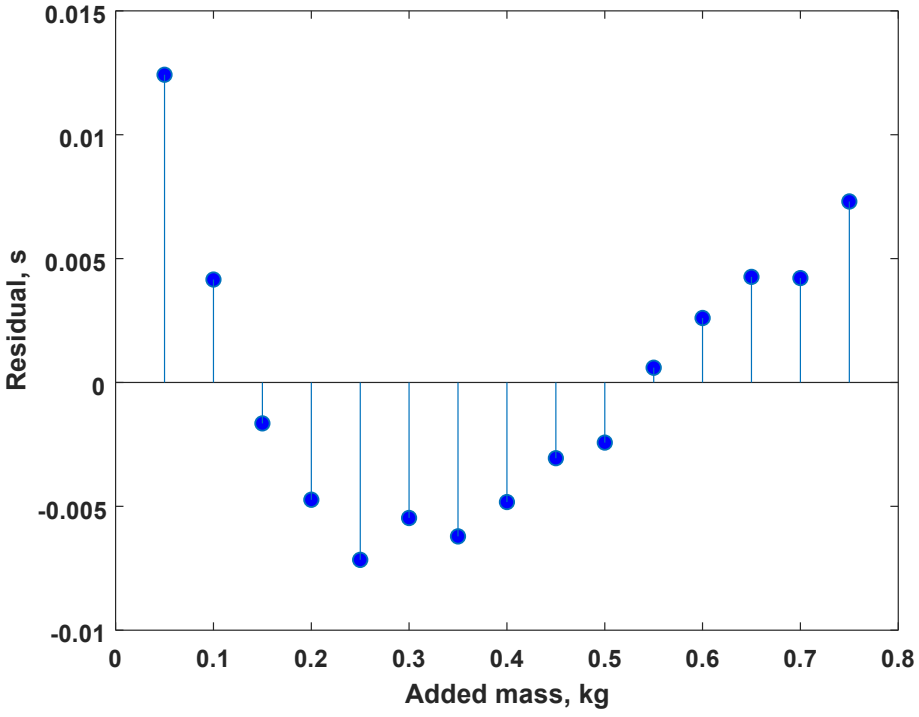

FIGURE 14

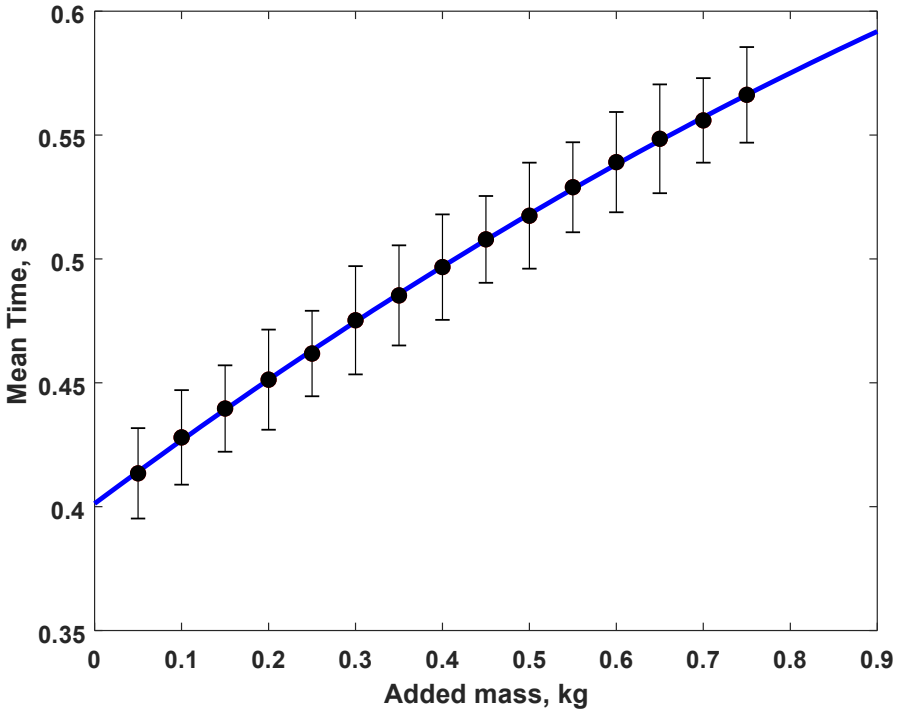

FIGURE 15 (Method 7)

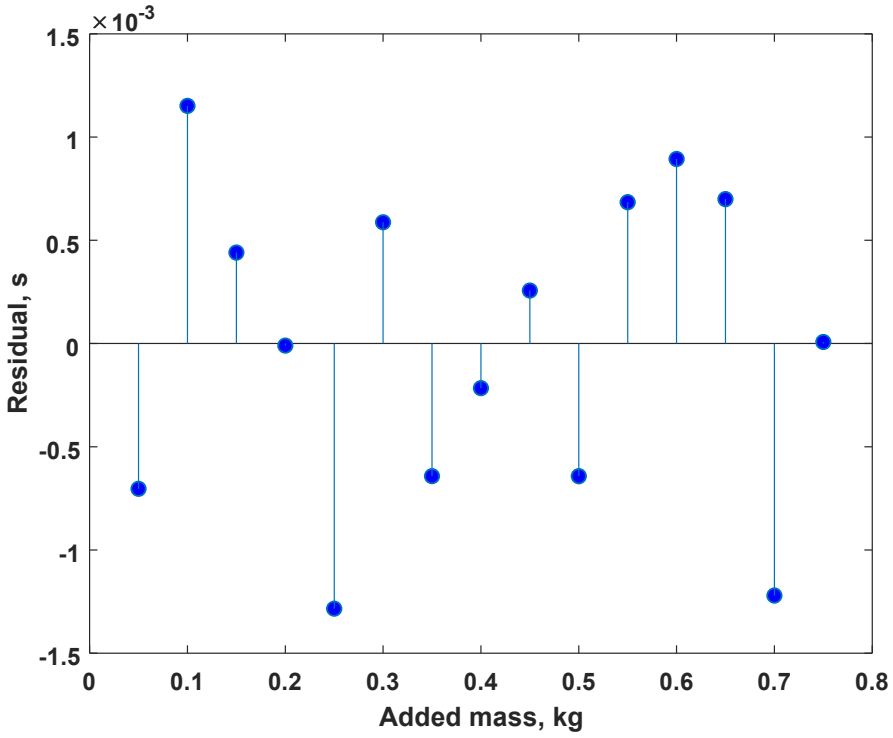

FIGURE 16