JOURNAL OF CHEMICAL EDUCATION

Article

pubs.acs.org/jchemeduc

Using Least Squares To Solve Systems of Equations

Joel Tellinghuisen*

Department of Chemistry, Vanderbilt University, Nashville, Tennessee 37235, United States

Supporting Information

ABSTRACT: The method of least squares (LS) yields exact solutions for the adjustable parameters when the number of data values n equals the number of parameters p. This holds also when the fit model consists of m different equations and $m = p$, which means that LS algorithms can be used to obtain solutions to systems of equations. In particular, nonlinear LS solves systems of nonlinear equations. An important example in chemistry is the case of reagents whose concentrations are coupled through multiple equilibrium relations. The capability of nonlinear LS in this application is examined for three programming environments, Excel Solver, FORTRAN, and KaleidaGraph, on a number of equilibrium problems having up to 10 unknown concentrations. FORTRAN and KaleidaGraph perform well in all the examples, but Solver presents difficulties that render it inadequate in several cases unless the problem is reformulated in terms of a smaller number of adjustable concentrations. When the input quantities (equilibrium constants, prepared concentrations) have uncertainty, the calculations can also be used to propagate these uncertainties into the derived quantities.

10^{-6} M bicarbonate in water

E1 = (1.e-6 – a – b – c); [a = HCO_3^-, b = CO_3^{2-}, c = H_2CO_3]

E2 = (2*b + a + f – d – 1.e-6); [d = H_3O^+, f = OH^-]

E3 = (1.e-14 – f*d);
E4 = (4.3e-7*c –a*d);
E5 = (4.8e-11*a – b*d);
(x==0)?(E1):((x==1)?(E2):((x==2)?(E3):((x==3)?(E4):(E5))));

KEYWORDS: *Upper-Division Undergraduate, Graduate Education/Research, Physical Chemistry, Laboratory Instruction, Analytical Chemistry, Problem Solving/Decision Making, Equilibrium, Acids/Bases, Chemometrics*

Practically every scientist and every undergraduate science major knows about the method of least squares (LS) for fitting data to a straight line. Most also understand that the LS solution is obtained when the number of data points n exceeds 2, yielding *overdetermined* equations. With just two (different) data points, the solution becomes an exact algebraic solution. This example involves a single equation, $y = a + bx$. The same considerations apply to other equations, linear and nonlinear, having p adjustable parameters: The LS solution occurs when $n > p$, an exact solution (in principle) when $n = p$, and an infinite number of solutions when $n < p$.

Probably fewer scientists appreciate that the same considerations apply when the fit model includes two or more equations. As a simple example, some of the data (n_1 points) might be fitted to one straight line, the rest (n_2) to a second straight line, giving $p = 4$. As long as n_1 and n_2 are both >2, we obtain LS solutions for both straight lines, with parameters identical to those that are obtained from separate fits. However, in general, the estimates of the parameter standard errors (SE) are different, because the pooled sum of squared residuals (SSQ) is used to scale the parameter variances in the combined fit in place of the separate SSQs for the individual fits.[1]

More interesting are cases where the multiple equations in the fit model share some parameters, for example, a common intercept or a common slope in the fit of data to two or more straight lines. We can obtain multiple estimates of the common parameters from separate fits and then average, but the statistically optimal approach is to estimate them from a single, combined or global fit.[2] In the two-straight-lines example, a common parameter reduces p to 3, and we obtain a LS solution as long as both n_1 and $n_2 \geq 1$ and $n_1 + n_2 > 3$. We obtain an exact solution when one of these n's is 1 and the other 2.

Taking this case to the extreme, suppose there are m equations in the model, with $m > p$. We again obtain a LS solution as long as there are data values for all m equations. An example from the physical chemistry teaching literature is the van der Waals equation for a gas, which has two parameters that can be obtained from three equations for the behavior at the critical point. Just two of these are used for an exact solution,[3,4] but all three can be used in a LS solution.[5]

Ordinarily, we think of the method of least squares as a way to estimate unknown parameters and their uncertainties from data in situations where either $n > p$ or (much less commonly) where $m > p$, giving statistical degrees of freedom $\nu = n - p$ or $m - p > 0$. However, common LS algorithms work fine for $\nu = 0$, and I have shown recently how LS can then be used as a tool for error propagation.[6] My illustrations were confined to the usual situation where there is a single equation in the fit model. Here, I extend those considerations to cases where the number of equations exceeds one, leading to systems of equations, linear and nonlinear, in which the number of equations equals the number of unknowns. Important chemistry examples include coupled equilibria involving acids and bases, limited solubility, and complexation. My emphasis is on obtaining the solutions to such systems; however, where uncertainty can be ascribed to the input quantities, the same calculations can yield propagated uncertainties in the derived quantities.[6]

Received: January 13, 2016
Revised: April 8, 2016

DOI: 10.1021/acs.jchemed.6b00027
J. Chem. Educ. XXXX, XXX, XXX−XXX

Methods for numerical solution of systems of equations include one-step procedures like matrix inversion for linear systems, and iterative methods, like Newton−Raphson, successive approximation, and bisection for nonlinear systems.[7] Most of these have been discussed in this *Journal*,[8−17] and have been implemented with pocket calculator strategies[13,15] and spreadsheets, especially Microsoft Excel.[10,14,16,17] The successive approximation methods work best when reduced to a single equation in one unknown, as they are sensitive to initial guesses for multiple variables and equations. The spreadsheet approaches mostly employ Excel's Solver routine and are not so limited; however, the methods for coupled equilibria have emphasized ways to reduce the number of equations to be solved. Here, I will use nonlinear LS (NLS) algorithms to obtain concentrations for all m species directly from the m equations defining the equilibrium relations, and mass and charge balance, the most basic relations describing the equilibria. To implement this approach, one must be able to fit different data points to different equations, and that requires slightly different coding in different programming environments. Accordingly, I will illustrate using Excel Solver, KaleidaGraph (Synergy Software),[18] and FORTRAN, with comparisons of their performance, in particular their convergence behavior for initial values far from the solutions, which is an indicator of the effort required to obtain correct results.

BACKGROUND

The method of least squares obtains solutions for unknown parameters by minimizing the sum of weighted squared residuals,

$$\mathcal{S} = \Sigma w_i \delta_i^2$$

where δ_i is the calculated-observed residual for the ith point, and w_i is the weight (= 1 for unweighted fitting, $\propto 1/\sigma_i^2$ for minimum-variance estimation when the data vary in precision[1]). For application to the solution of systems of equations, each i must refer to a different equation, with the number of equations equaling the number of data points. On solution, each δ_i is exactly zero, giving $\mathcal{S} = 0$. However, in practice, the computational methods are limited in precision and will converge on just a very small value for $\mathcal{S}$. For systems of linear equations, convergence occurs (in principle) in a single computational step; however, the more interesting nonlinear applications require iteration from a set of initial estimates provided for the unknowns.

The idea of using least squares to solve equations is not new. Press et al.[7] mention it briefly near the end of Chapter 9 but dismiss it because of concerns about the problem of multiple local minima, a known (but usually manageable) problem of nonlinear LS. However, there are powerful methods for finding these minima, especially the Levenberg−Marquardt algorithm,[7,19] which ensures that *some* minimum will be found. And knowledge that the correct solution should yield $\mathcal{S} = 0$ makes it easy to recognize incorrect solutions associated with local minima, after which one can revise the initial estimates and try again. In fact, Christian and Tucker long ago provided a general purpose program (SEQS) implementing this approach.[20,21] Today it is easy to do the same with microcomputer spreadsheet and data analysis programs, as I will illustrate.

The residual δ_i in the sum of squares $\mathcal{S}$ is the mismatch between calculated and observed values for the ith point. For the KaleidaGraph program, one must prepare an x−y plot before executing a fit, and because of that it is convenient to write all equations in the form $f = 0$, with the plotted y values ("observed") all being 0. For example, the equation, $a + bx = 3$ is written as $f = a + bx - 3 = 0$. There can still be computational difficulties when the typical magnitudes of the δ_i vary widely, so, e.g., in a particular case, this equation might better be expressed as $g = 1 + bx/a - 3/a = 0$.

ILLUSTRATIONS

Roots of Polynomials

Consider first the simplest problems, involving a single equation. Bamdad has treated 11 different equilibrium problems that can be expressed as a cubic polynomial in a single unknown, and has shown results for three different successive approximation algorithms and a bisection routine on a pocket calculator.[15] Only the last of these yielded results in all 11 cases. I will illustrate the LS approach on his seventh example, where two of the successive approximation methods diverged.

This example involves HCN ionization in water, for which there are 4 equations for 4 unknown concentrations, $[H_3O^+]$, $[CN^-]$, $[HCN]$, and $[OH^-]$, units mol/L:

$$K_a = [H_3O^+][CN^-]/[HCN] = 6.2 \times 10^{-10} \quad (1a)$$

$$K_w = [H_3O^+][OH^-] = 1 \times 10^{-14} \quad (1b)$$

$$[CN^-] + [HCN] = [HCN]_0 = 1 \times 10^{-6}\ \mathrm{M} \quad (1c)$$

$$[H_3O^+] = [CN^-] + [OH^-] \quad (1d)$$

The first two equations express the ionization equilibria for HCN and H_2O, the third is mass balance for CN species starting with 10^{-6} M HCN, and the fourth covers charge balance. Solving for $[H_3O^+]$ and representing it by a, we obtain the cubic equation,

$$a^3 + K_a a^2 - (K_w + [HCN]_0 K_a)a - K_a K_w = 0 \quad (2)$$

To solve eq 2 in Excel with the LS method, we might use the spreadsheet illustrated in Figure 1. The constants are stored in

Example1-M.xls

	A	B	C	D	E
1	quantity		conc	eqns	SSQ
2					
3	[H3O+]	-6.98701	1.0303539E-07	-1.32E-06	1.74E-12
4	C0	1.00E-06			
5	Ka	6.20E-10			
6	Kw	1.00E-14			
7					

```
C3=10^B3
D3=(C3^3 + B5*C3^2 - (B6+B4*B5)*C3 - B5*B6)*1E+27
E3=(D3)^2
```

Figure 1. Excel spreadsheet for the root of eq 2 ($[H_3O^+]$, in mol/L), showing converged results from minimizing E3 with Solver, by varying B3. Precision is for numerical purposes only.

B4:B6, and cells C3:E3 are populated using the equations shown. The concentration (C3) is expressed in terms of its logarithm in B3, a useful device for keeping the concentration positive.[17] The sum of squares is a single term in E3, which becomes the minimization target for Solver, with B3 being the quantity varied. The quantities in row 3 are converged results.

DOI: 10.1021/acs.jchemed.6b00027
J. Chem. Educ. XXXX, XXX, XXX−XXX

Note that eq 2 has been scaled by 10^{27} in D3. Without significant scaling, Solver quits iterating before achieving the solution, evidently a manifestation of its known difficulty dealing with very small numbers (Carl Salter, private communication). Even with such scaling, this routine converges for only a small range of values of B3, from −7.2 to −6.6; outside this range, it jumps to C3 = 0 and takes this as the solution. (There is another root at C3 = -5.8×10^{-10}, which is inaccessible to C3 with its exponential definition.) Changing units to millimolar (mM), micromolar (μM), and nanomolar (nM) (in which, e.g., K_w becomes larger by factors of 10^6, 10^{12}, and 10^{18}, respectively) reduces the need for scaling but does not expand the range of B3 values for convergence. Nor does constraining B3 help.

Since here there is only a single contribution to the sum of squares in E3, one can alternatively use Solver in its root-finding mode, targeting D3 for the value of C3 that yields D3 = 0. In this mode, I achieved correct convergence for B3 values ranging from −7.2 to 3.2. However, at the upper end of this range, Solver sometimes claimed to have found a solution when it actually had not, or contrariwise, it sometimes pronounced "no solution" when it *had* converged on the correct value. Also, sometimes it required a second pass, after stopping on a value close to the answer. I could find no simple connection between such behavior and values entered for Solver's Options.

To solve this problem in KaleidaGraph (KG), we use its General routine. This is an example of a NLS algorithm where the user can specify the fit function. Simple fit models (e.g., a + b*x for a straight line) can be entered directly in the Define box that opens when General is selected under the Curve Fit menu; for more complex models it is convenient to use the Macro Library. To execute a fit, we must first prepare an x–y plot, and in so doing, we must also deal with a nuisance quirk of the KG program, a requirement for at least three data points before it will proceed to a solution. I discussed this problem in ref 6, in which I noted that this limitation is not logical, in that it applies irrespective of the number of adjustable parameters. For the present example we enter the values 0, 1, 2 in rows 0–2 of the first column (c0) in the data sheet (our x, needed for plotting but values unimportant here), and zeroes beside them in column c1. The latter will be our y values, as we will write all equations in the form $f(x) = 0$. We pick Scatterplot under the Gallery menu and select the first column for x and the second for y, producing the plot from which we will invoke the fit. We then select a fit model name from the General submenu under Curve Fit and obtain results illustrated in Figure 2, which includes a propagated error in a, obtained as described below. Here, the quantity to the right of the = sign in the results box has been entered in the Define box, and the quantities Ka, Kw, and C0 have been assigned numerical values in the Macro Library. Correct convergence occurred for initial values of a as much as 16 orders of magnitude too large, but only for a factor of 2 too small. Fast (~1 s) convergence was obtained for all 11 examples in ref 15 using the initial value a = 1.

The propagated error in $[H_3O^+]$ in Figure 2 was obtained for assumed $\sigma = 1 \times 10^{-11}$ for K_a, using a weighted fit.[6] For this, we need σ_y, which we obtain by propagating the error in K_a into y. K_a occurs in all but the first term in eq 2, with the dominant variance contribution coming from the third term; the propagated error in y is 1.0242×10^{-24}, which we enter in the c2 column ("sig") in the data sheet. But this is for a single value, and we cannot use it for all 3 values, as that will overweight the data. This problem is discussed in ref 6 and here in the Supporting Information. Figure 2 shows one way to deal with it: greatly downweight two of the three values by assigning them large σ values. In this case, with only K_a considered uncertain, we can obtain the same results by rerunning the fit of Figure 2 with K_a altered by its uncertainty, e.g., using $K_a = 6.1 \times 10^{-10}$, and noting the resulting change in a.

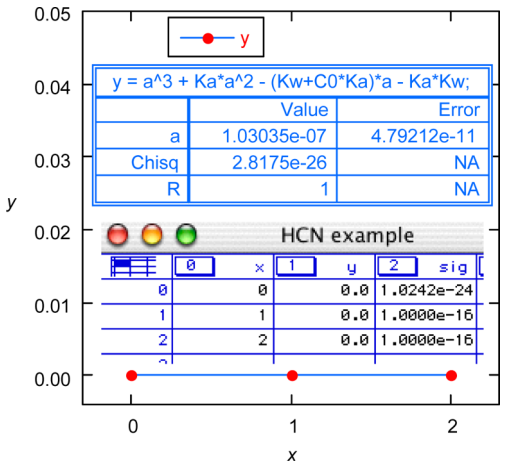

Figure 2. KaleidaGraph solution for the root of eq 2 ($[H_3O^+]$, in mol/L), using the General least-squares routine. Also shown is the relevant portion of the KG data sheet. The Error for a is the propagated uncertainty for an uncertainty of 1×10^{-11} in K_a.

Multiple Equations

Consider a system of 3 linear equations in 3 unknowns. The Excel spreadsheet counterpart to Figure 1 might now contain the 3 unknowns in C3:C5 and the 3 equations in D3:D5, in place of the single entries in C3 and D3. Then, E3 would contain "=D3^2 + D4^2 + D5^2" and would be the target for minimization by varying C3:C5. The operations equivalent to these are handled automatically in NLS algorithms, but they require that the program associate the 3 equations with 3 different data points. Figure 3 shows how this can be done with KG. Note that now the values of x are important, as they are used to direct the algorithm to the different functions through the shown nested conditional operator (?:), common to the C programming languages. Accordingly, the three data points (all $y = 0$) must have x values 0, 1, and 2 (or any value ≠ 1 or 2). For comparison, a FORTRAN program might accomplish this using the 3 lines at the bottom of the figure.

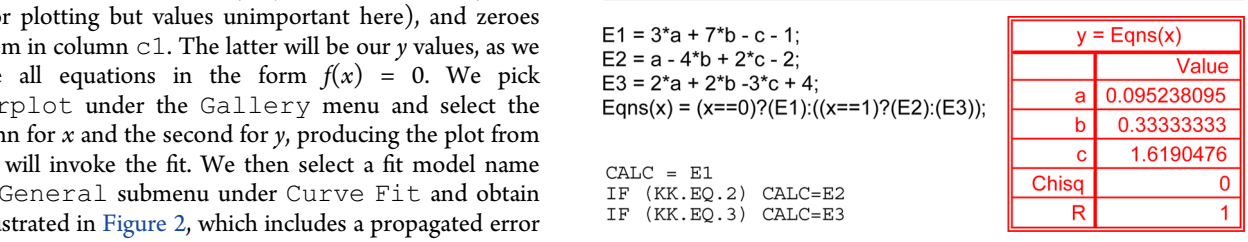

Figure 3. KaleidaGraph solution of 3 equations in 3 unknowns. The 4 lines at top are entered in the Macro Library, and "Eqns(x)" is entered in the Define box. The data sheet from Figures 2 and 3 can be used again here. The KG Eqns(x) statement might be replaced by the 3 lines at bottom in the FUNCTION routine of a FORTRAN program like Program 11-2 in ref 19 (with the Ei defined the same). KK is the index (1–3) of the data points.

DOI: 10.1021/acs.jchemed.6b00027
J. Chem. Educ. XXXX, XXX, XXX–XXX

The equations in this example are linear, so they can be solved in a single step using matrix methods or determinants. Next, we attack the equilibrium problem of Figures 1 and 2 from the bare-essentials approach: 4 equations, some nonlinear, in 4 unknowns. To accomplish this, we extend the ideas in Figure 3, as illustrated in Figure 4. For the KG fit, we use the

```
Ka = 6.2E-10;
Kw = 1.E-14;
C0 = 1.E-6;
H = m1;
CN = m2;
HCN = m3;
OH = m4;
E1 = (Ka - H*CN/HCN);
E2 = (Kw/H - OH);
E3 = (HCN + CN - C0);
E4 = (H - CN - OH);
Eqns(x) = (x==0)?(E1):((x==1)?(E2):((x==2)?(E3):(E4)));
```

y = Eqns(x)	
	Value
H	1.03035e-07
CN	5.98136e-09
HCN	9.94019e-07
OH	9.7054e-08
Chisq	0
R	1

HCN-e

	A	B	C	D	E
1	species		conc	eqns	SSQ
2	H+	-6.98701	1.0304E-07	-3.28E-15	1.22E-15
3	CN-	-8.22320	5.9814E-09	-6.98E-16	
4	HCN	-6.00261	9.9402E-07	9.95E-16	
5	OH-	-7.01299	9.7054E-08	4.02012E-17	

```
D2 = 6.2E-10 - C2*C3/C4
D3 = 1.E-14/C2 - C5
D4 = C4 + C3 - 1.E-6
D5 = C2 - C3 - C5
```

Figure 4. Treatment of HCN ionization equilibria for $[HCN]_0 = 10^{-6}$ M using KaleidaGraph (top) and Excel Solver. The equations at top are entered in the KG `Macro Library`; the first 7 define constants and assign the 4 variables to `m1`–`m4`, and the next 4 express eqs 1. The fit model `Eqns(x)` uses the conditional operator to designate `E1` for the first data point, `E2` for the second, etc., and is entered in the `Define` box. The same 4 equations are entered in `D2:D5` of the Excel worksheet, as shown at bottom. The target for minimization `E2` is the sum of the squares of these equations scaled by 10^{14}. Here, the concentrations (in `C2:C5`) are determined by their logarithms in `B2:B5`, e.g., by `C2 = 10^B2`.

same data sheet in Figure 2 by just adding a fourth row having $x = 3$. This fit yields nearly instantaneous results for initial values between 10^{-15} and 10^{6} for all 4 concentrations (see Supporting Information for details and error propagation).

The Excel Solver solution required the indicated large scale factor in `E2` to avoid the small numbers problems. However, with this (and even larger scaling), the computations did converge correctly over at least 14 orders of magnitude variation in the initial concentrations. For initial values more than 2 orders of magnitude larger or smaller than the final values, the program required two or even three passes to achieve true convergence. In these cases, it incorrectly claimed to have found a solution after the first (and/or second) pass, but rerunning from that point produced further reduction in SSQ and eventual convergence.

By using the method of eqs 1 and Figure 4, we have avoided the question, "where does the H_3O^+ come from?" The stoichiometry approach assumes that there are two contributions to $[H_3O^+]$, say α and β, from the two ionization equilibria. Accordingly, β = the present $[OH^-]$ and $\alpha = [H_3O^+] - [OH^-] = [CN^-]$. The present approach can be even more advantageous for polyprotic acids. Consider the dissolution of HCO_3^- in water. Now we have an additional ionization equilibrium and an additional unknown, giving 5 equations in 5 unknowns. Three of these are the equilibrium expressions for K_w, K_{a1}, and K_{a2}. The other two are mass balance for the CO_3 species and charge balance:

$$[HCO_3^-] + [H_2CO_3] + [CO_3^{2-}] = [HCO_3^-]_0 \quad (3a)$$

$$[HCO_3^-]_0 + [H_3O^+] = [HCO_3^-] + 2[CO_3^{2-}] + [OH^-] \quad (3b)$$

where the first term on the l.h.s. of eq 3b is the contribution from the cation that accompanies HCO_3^- in the solute. For $[HCO_3^-]_0 = 0.1$ M, the resulting concentrations for the 5 species span a range $>10^7$, and the computations can be sensitive to the choice of initial values. Using the same initial value for all species, I initially achieved convergence with KG for only the narrow range 10^{-3}–10^{-1} M. However, through judicious use of scale factors and ways of expressing the equilibrium relations, I could expand the convergence range to initial values from 10^{-10} M to 10 M (details in Supporting Information). Figure 5 shows results obtained for this bicarbonate concentration and for 10^{-6} M.

$[HCO_3^-]_0 = 0.1$ M

y = (x==0)?(E1):((x==1)?(E2)...	
	Value
$[HCO_3^-]$	0.097931
$[CO_3^{2-}]$	0.0010336
$[H_2CO_3]$	0.0010358
$[H_3O^+]$	4.5479e-09
$[OH^-]$	2.1988e-06
Chisq	0
R	1

10^{-6} M

y = (x==0)?(E1):((x==1)?(E2)...	
	Value
$[HCO_3^-]$	8.8162e-07
$[CO_3^{2-}]$	7.3752e-10
$[H_2CO_3]$	1.1764e-07
$[H_3O^+]$	5.7378e-08
$[OH^-]$	1.7428e-07

Figure 5. KaleidaGraph results for dissolving HCO_3^- in water at indicated total concentrations (all concentrations in M). The calculations used $K_{a1} = 4.3 \times 10^{-7}$ and $K_{a2} = 4.8 \times 10^{-11}$.

As is discussed in the Supporting Information, this example proved demanding for my FORTRAN code, also. Without scaling of the equilibrium relations for K_w and K_{a2}, my Marquardt routine often converged to local minima associated with negative concentrations, and a simpler routine that employed scale factors to keep all concentrations positive often produced fatal numerical errors. Scale factors of 10^5 and 10^3 for the K_w and K_{a2} equilibria, respectively, yielded correct convergence over a wide range of starting concentrations.

The 0.1 M bicarbonate example was even more difficult to treat using Excel Solver. Working in molar (M) concentration units, I could obtain nearly correct results for the 3 large concentrations but not the two smaller ones, even starting

DOI: 10.1021/acs.jchemed.6b00027
J. Chem. Educ. XXXX, XXX, XXX−XXX

within a factor of 2 of their correct values. Scaling the equations to change their relative significance did not help. Working in millimolar (mM) units and scaling the equilibrium equations for K_w and K_{a2} finally gave accurate concentrations for all 5 species. However, these results would have been difficult to achieve without prior knowledge of the correct answers. For these reasons, I conclude that Solver fails this test.

The KaleidaGraph program is limited to 9 adjustable parameters in its LS routines. Thus, with cases like the third example in ref 17, where there are 10 species, some reduction must be done in order to solve the problem in KG. An obvious choice in acid–base problems is to replace $[OH^-]$ with $K_w/[H_3O^+]$. KG also limits nesting in the conditional test statements used in Figures 3–5 to a depth of 5. However, one can define a two-step sort here, through the following Library definitions:

```
FeHQ(x) = (x<3)?(FeHQ1(x)):((x<6)?(FeHQ2(x)):(FeHQ3(x)));
FeHQ1(x) = (x==0)?(E1):((x==1)?(E2):(E3));
FeHQ2(x) = (x==3)?(E4):((x==4)?(E5):(E6));
FeHQ3(x) = (x==6)?(E7):((x==7)?(E8):(E9));
```
(4)

`FeHQ(x)` is then entered in the `Define` box of a `General` fit. Initial values of 0.01 for all concentrations instantly produced the results in Figure 6, which agree completely with those in ref 17, within the lower precision of the latter (details in Supporting Information).

y = FeHQ(x)	
	Value
Fe^{3+}	2.5178e-05
Fe^{2+}	0.019967
$Fe(Ac)^{2+}$	2.7426e-06
$Fe(OH)^{2+}$	5.0073e-06
HAc	0.049954
Ac^-	4.3366e-05
H_2Q	0.010016
H_3O^+	0.020018
Q	0.0099835
Chisq	1.1755e-38
R	1

Figure 6. Results for reduction of Fe^{3+} by hydroquinone in the presence of acetic acid, example 3 from ref 17. Concentrations in molar (M).

Comparison with Other Solver-Based Methods

Although the authors of ref 17 first mention "least squares" near the end of their paper, their Solver-based method is fundamentally the same as the present method, where I have followed their implementation in the Excel spreadsheets shown in Figures 1 and 4, including concentrations defined in terms of the adjustable exponents of 10 in column `B`. While the latter procedure ensures positive concentrations, my observations indicate it can also lead to slow convergence. Although the ref 17 authors have chosen to reduce the number of simultaneous equations by the number of equilibrium relations (expressing some unknown concentrations in terms of others), there is no fundamental reason for not retaining a different number of equations and unknowns for simultaneous solution. This point is illustrated with a specific example below (see also Supporting Information).

I have tested the ref 17 method on the demanding bicarbonate example in both KG and Solver, for $[HCO_3^-]_0$ = 0.1 M. When $[H_3O^+]$ and $[HCO_3^-]$ were taken as the optimization variables, I could not succeed with Solver but had no problem with KG. Note that these two concentrations represent the extremes in this system. When instead $[HCO_3^-]$ and $[H_2CO_3]$ were chosen, both programs worked well. These results support the contention that Solver does not perform well in small-number optimizations, and especially when the optimization variables vary substantially in magnitude. As further substantiation of this claim, for the problematic former case, a simple trick yielded success with Solver for initial values ranging over 15 orders of magnitude: Define the optimization variable for $[H_3O^+]$ as a quantity larger by a factor of 10^7, making it comparable to $[HCO_3^-]$. (Details can be found in the Supporting Information.)

De Levie's method in ref 16 also involves reducing the equations to a small number, just one in his first example (Cd^{2+}–Cl^- complexation), two in his second (Hg^{2+} with Cl^- and OH^-). This method requires some algebraic manipulation of the basic equations that many users may find nontrivial, so it is of interest to see how well the present approach works in these cases. There are 6 equations in 6 unknowns in the first example, and KG yielded correct results with no modification of the basic equations. However, for the second example (9 unknown concentrations, for Hg^{2+}, Cl^-, OH^-, 4 HgCl complexes, and 2 HgOH complexes), KG initially failed unless provided with initial values that were practically correct. But this example involves products of very large binding constants K and very small concentrations, and results in such cases can be sensitive to just how the program does the computations. A very simple change produced immediate success and confirmed that this large–small number situation was the problem: The equations were expressed in terms of the reciprocal K's (dissociation constants) instead of the K's, e.g., the equation for $[HgCl_2]$ was written as

```
E1 = HgCl2 - Hg*Cl^2/Ki2;
```
(5a)

instead of as

```
E1 = HgCl2 - K2*Hg*Cl^2;
```
(5b)

where `K2` is β_2 in Figure 3 of ref 16 (= $10^{13.22}$), and `Ki2` is its reciprocal. In other words, dividing by $10^{-13.22}$ works, while multiplying by $10^{13.22}$ fails, an example of how subtle details can spell success or failure in such computations. Changing to millimolar (mM) concentration units also yielded success. For example, in eq 5b, this reduces K_2 by a factor of 10^6 and increases $[Hg^{2+}][Cl^-]^2$ by 10^9, producing at least 9 orders of magnitude reduction in the extremes of the multipliers in eq 5b in molar (M) units. In millimolar units, KG converged correctly for initial values (all the same) from 10^{-6} to 300 mM. A non-Marquardt FORTRAN code that uses scale factors to keep all concentrations positive converged on correct results for initial concentrations from 0.003 to 100 mM. Excel's Solver did not succeed for any single initial concentration for all species (details in Supporting Information).

As has already been noted, one can always limit the specifically optimized variables to a chosen subset of the unknowns, while still adjusting the other values in each computational iteration. With its 9 unknowns, this last example is a good one on which to illustrate this point, since convergence difficulties generally decrease as the number of

DOI: 10.1021/acs.jchemed.6b00027
J. Chem. Educ. XXXX, XXX, XXX–XXX

unknowns is decreased. In KG, for example, in place of E1 in eq 5b, we can write

```
HgCl2 = K2*Hg*Cl^2;                                    (6)
```

to explicitly compute $[HgCl_2]$ from $[Hg^{2+}]$ and $[Cl^-]$. We then remove HgCl2 from the list of optimized parameters and E1 from the equations. Doing this for all complexes *other than* $HgCl^+$ and $HgCl_2$ reduces the number of equations and unknowns to 5, and with this model, I obtained convergence in KG working in molar concentration units, for initial concentrations (all the same) from 10^{-7} to 10^{-2} M. Doing the same for $HgCl^+$ and $HgCl_2$ reduces the equations and unknowns to 3, equivalent to the method of ref 17. With the three balance equations being mass balanced for Hg, Cl, and OH, I found that the convergence range was actually reduced for KG, from 10^{-6} to 10^{-3} M. Further equation reduction requires algebraic solution for one of these three concentrations in terms of the others, for example, solving either the Hg mass balance equation or the OH balance for $[Hg^{2+}]$. The former choice gave a comparable convergence range, 10^{-6} to 10^{-1} M; the latter gave expanded convergence, 10^{-10} to 10^{-3} M, but requiring many more iterations. My efforts with Solver, working in millimolar units, yielded complete success only for 2 equations in $[Cl^-]$ and $[OH^-]$, with the Hg balance equation used to solve for $[Hg^{2+}]$. This is equivalent to the approach in Figure 3 of ref 16, except that only the expression for $[Hg^{2+}]$ requires algebra beyond the definitions like eq 6. (More details can be found in the Supporting Information.)

CONCLUSION

I have examined several nonlinear least-squares programs for their ability to solve systems of coupled chemical equilibrium equations in their most basic form, with each of p unknown concentrations taken as an adjustable parameter obtainable by minimizing the sum of squared residuals of the corresponding p equilibrium relations. This approach is arguably the simplest mathematically and the easiest to code for computation. For the two desktop programs compared here, Excel Solver and KaleidaGraph, the latter performed much more reliably, yielding correct results over much larger ranges of input initial values for the concentrations. Solver's problems in dealing with very small numbers and with adjustable parameters that span a large magnitude range are a major drawback that can sometimes be reduced to an inconvenience. But in several of the present test cases, they prevented success unless the initial values were unreasonably close to the solutions. Solver's frequent misreporting of its iteration results is more dangerous, potentially leading users to accept incorrect results. The in-house FORTRAN codes tested here were generally comparable to KG in performance.

The number of optimized LS concentrations and equations can always be reduced by using some of the equations to relate concentrations, for example, removing $[OH^-]$ and the K_w equation by replacing $[OH^-]$ with $K_w/[H_3O^+]$ in acid–base problems. The concentrations so removed from the parameter list are still adjusted iteratively in the computations, so their correct values are available when the computations converge on a solution. Such procedures are necessary for KaleidaGraph with problems having more than 9 unknown concentrations, as KG is limited to 9 LS parameters. Such reduction procedures are at the heart of the methods of refs16 and 17, in both of which Solver was used successfully to obtain the solutions. However, in one of the present examples, dissolution of bicarbonate, special scaling procedures were needed for success with Solver in the method of ref 17, even though the original problem had been reduced from 5 unknowns and equations to just two. Similarly, Solver's performance in the tests on the Figure 3 example from ref 16 was greatly limited compared with that for KaleidaGraph.

Solver's "no convergence" results are a manageable problem, because a user will almost always try another pass at the problem. The false convergence reports are a more serious concern. These appear to be cases where the Solver Marquardt routine "hangs up" on a local near-minimum. I checked this possibility on several occasions by taking as initial values for the KG version of the same problem the "converged" Solver results, copied to 11 significant figures. KG always proceeded to a correct solution in such tests, though sometimes requiring tens of thousands of iterations! (This test is not perfect, because there are still differences in machine accuracy and 11 decimal digits, but it does substantiate the local minimum explanation.) Solver could sometimes be "jogged" to converge further by altering one or more equations or changing their significance in the sum of squares through scale factors, but this is not a practically useful approach for most users; nor can it be relied upon to work in all cases.

It is interesting that the present all-equations approach arises more naturally in the Solver environment than it does for NLS algorithms like KaleidaGraph's General routine. In the latter, one must employ unorthodox methods to associate each data point with a different equation. In Excel on the other hand, the different equations are entered in cells in the spreadsheet, and the sum of their squares is straightforwardly designated as a target for minimization by Solver. It is easy to imagine that the present approach has been tried previously by Solver users and then abandoned because of Solver's numerical limitations. The present tests show clearly that those limitations are not "ordained by nature". In view of the widespread use of Solver for data analysis by many scientists and educators, it would be good if the maintainers of the Solver program could be persuaded to fix its problems.

ASSOCIATED CONTENT

Supporting Information

The Supporting Information is available on the ACS Publications website at DOI: 10.1021/acs.jchemed.6b00027.

Computational details for several of the examples treated in the text, as well as for error propagation and for working in different concentration units; more extensive comparison tests of Solver vs KaleidaGraph and FORTRAN (PDF)

AUTHOR INFORMATION

Corresponding Author

*E-mail: joel.tellinghuisen@vanderbilt.edu.

Notes

The author declares no competing financial interest.

REFERENCES

(1) For weighted fits in which the SEs are taken from the a priori covariance matrix, the SEs will be the same in the two cases, but the χ^2 values will be different. See Tellinghuisen, J. *J. Chem. Educ.* **2005**, *82*, 157–166.

DOI: 10.1021/acs.jchemed.6b00027
J. Chem. Educ. XXXX, XXX, XXX–XXX

(2) Salter, C. A Global Least-Squares Fit for Absolute Zero. *J. Chem. Educ.* **2003**, *80*, 1033−1035.
(3) Levine, I. N. *Physical Chemistry*, 6th ed.; McGraw-Hill: New York, 2009.
(4) Eberhart, J. G. The Many Faces of van der Waals's Equation of State. *J. Chem. Educ.* **1989**, *66*, 906−909.
(5) Eberhart, J. G. A Least-Squares Technique for Determining the van der Waals Parameters from the Critical Constants. *J. Chem. Educ.* **1992**, *69*, 220−221.
(6) Tellinghuisen, J. Using Least Squares for Error Propagation. *J. Chem. Educ.* **2015**, *92*, 864−870.
(7) Press, W. H.; Flannery, B. P.; Teukolsky, S. A.; Vetterling, W. T. *Numerical Recipes*; Cambridge University Press: Cambridge, U.K., 1986.
(8) Stone, E. E. Complex Chemical Equilibria. *J. Chem. Educ.* **1966**, *43*, 241−244.
(9) Olivieri, A. C. Solution of Acid-Base Equilibria by Successive Approximation. *J. Chem. Educ.* **1990**, *67*, 229−231.
(10) Carter, D. W.; Frye, M. S.; Mattson, W. A. A Direct Approach for Solving Simultaneous Equations — Combining a Basic Understanding of Equilibria with the Convenience and Power of a Spreadsheet. *J. Chem. Educ.* **1993**, *70*, 67−71.
(11) Eberhart, J. G. Solving Nonlinear Simultaneous Equations by the Method of Successive Substitution. *J. Chem. Educ.* **1994**, *71*, 1038−1040.
(12) Uribe, D. Solving Chemical Equilibria. *J. Chem. Educ.* **1998**, *75*, 1177−1179.
(13) Donato, H., Jr. Graphing Calculator Strategies for Solving Chemical Equilibrium Problems. *J. Chem. Educ.* **1999**, *76*, 632−634.
(14) Guinon, J. L.; Garcia-Anton, J.; Perez-Herranz, V. Spreadsheet Techniques for Evaluating the Solubility of Sparingly Soluble Salts of Weak Acids. *J. Chem. Educ.* **1999**, *76*, 1157−1160.
(15) Bamdad, F. Solution of Cubic Equations by Iteration Methods on a Pocket Calculator. *J. Chem. Educ.* **2004**, *81*, 758−761.
(16) de Levie, R. How to Compute Labile Metal-Ligand Equilibria. *J. Chem. Educ.* **2007**, *84*, 136−141 There appear to be some errors in Figure 3 and its caption: (a) $\log \beta_2 = 13.22$; (b) in the equation in B13, B16 should be B6; (c) in the equation in E2, E9 should be B10.
(17) Baeza-Baeza, J. J.; Garcia-Alvarez-Coque, M. C. Systematic Approach to Calculate the Concentration of Chemical Species in Multi-Equilibrium Problems. *J. Chem. Educ.* **2011**, *88*, 169−173.
(18) Tellinghuisen, J. Nonlinear Least-Squares Using Microcomputer Data Analysis Programs: KaleidaGraph in the Physical Chemistry Teaching Laboratory. *J. Chem. Educ.* **2000**, *77*, 1233−1239.
(19) Bevington, P. R.; Robinson, D. K. *Data Reduction and Error Analysis for the Physical Sciences*, 2nd ed.; McGraw-Hill: New York, 1992.
(20) Christian, S. D.; Tucker, E. E. Least Squares Analysis with the Microcomputer: Solving Simultaneous Equations *American Laboratory* **1983**, *15*, 78−85.
(21) Dierenfeldt, K. E. SEQS 3.0 Student Version Simultaneous Equation Solver (Tucker Edwin E.). *J. Chem. Educ.* **1990**, *67*, A149.

DOI: 10.1021/acs.jchemed.6b00027
J. Chem. Educ. XXXX, XXX, XXX−XXX