

Refining the General Symmetric Definite Eigenproblem

§0: Introduction

Given are two symmetric n -by- n matrices $A = A^T$ and $H = H^T$; and H is *Positive Definite*. The *Eigenproblem* “ $A \cdot \mathbf{e} = H \cdot \mathbf{e}$ ” has n real eigenvalues and eigenvectors $\mathbf{e} \in \mathbb{R}^n$. Each eigenvalue is a zero of the *Characteristic Polynomial* $\det(\lambda H - A)$ and also a *Stationary Value* of the *Rayleigh Quotient* $R(\mathbf{x}) := \mathbf{x}^T \cdot A \cdot \mathbf{x} / \mathbf{x}^T \cdot H \cdot \mathbf{x}$ for $\mathbf{x} \in \mathbb{R}^n$ since only if \mathbf{x} is an eigenvector can the derivative $R'(\mathbf{x}) = 2(A \cdot \mathbf{x} - H \cdot \mathbf{x} \cdot R(\mathbf{x}))^T / \mathbf{x}^T \cdot H \cdot \mathbf{x}$ vanish, and then $R(\mathbf{x})$ is \mathbf{x} 's eigenvalue. Consequently $R(\mathbf{e} + \delta \mathbf{e}) = R(\mathbf{e}) + O(\|\delta \mathbf{e}\|^2)$ as $\mathbf{e} + \delta \mathbf{e}$ approaches an eigenvector \mathbf{e} whose eigenvalue is λ . This facilitates the refinement of approximate eigenvalues given approximate eigenvectors. Approximate eigenvectors are harder to refine.

A well-known way to refine an estimated eigenvector is *Inverse Rayleigh Quotient Iteration*:

Choose an eigenvector estimate $\mathbf{f}_0 \in \mathbb{R}^n$;

For $k = 0, 1, 2, 3, \dots$ in turn do ...

$$\{ \mathbf{u}_k := (A - (\mathbf{f}_k \cdot H)^{-1} \cdot H \cdot \mathbf{f}_k); \mathbf{f}_{k+1} := \mathbf{u}_k / (\mathbf{u}_k^T \cdot H \cdot \mathbf{u}_k) \}.$$

In the late 1950s Alexandre Ostrowski proved that a sequence $\pm \mathbf{f}_k$ converges *Cubically* to an eigenvector \mathbf{e} if \mathbf{f}_0 is close enough to it. In the late 1960s I proved that convergence to some eigenvector occurs from every choice of \mathbf{f}_0 except those in a repulsive set of measure zero; see §§4.6 - 4.9 of B.N. Parlett's (1998) book. But Rayleigh quotient iteration takes too long, $O(n^4)$, to refine all n eigenvectors. There seems to be a faster way; it is the main subject of this paper.

The faster way is an iteration analogous to Jacobi's iteration used when the eigenproblem's H is the identity matrix I ; see Ch. 9 of Parlett's book. When $H = I$ analogous iterations have been used for about four decades by structural engineers to compute vibrational modes and frequencies of undamped elastic structures; see Ch. 15 of Parlett's book, and Slapničar & Hari (1991). Engineers have used similar iterations despite that their convergence had not been proved unless started close enough. A proof that this paper's iteration converges globally is outlined in §6.

This paper's and other analogous iterations are algorithms that admit several implementations as computer programs not all of which are numerically stable in the face of rounding errors. The instability of the most obvious implementations was diagnosed with the aid of a debugging tool unavailable to almost all other programmers. This tool's prowess is displayed in §3. A stable reformulation derived in §2 is embodied in a MATLAB program `gnsym eig.m` listed in §11.

MATLAB's own `eig(A, H)`, like every other program intended to solve the general symmetric definite eigenproblem, is susceptible to a rarely occurring failure mode induced by roundoff and difficult to detect. The failure can occur only when vectors exist that are too nearly annihilated by both A and H . When this failure occurs, albeit very rarely, current computing practice most likely lets it pass unnoticed; we cannot know how often. How to detect such a failure and how to remedy it aided by iterative refinement of eigenvectors is discussed with an example in §8. Of course, iterative refinement's intricacies would have little value if fast floating-point hardware of extravagantly high precision were commonplace, but that might not come about anytime soon.

§1: A Characterization of Eigenvectors and their Pathologies

Let E be a matrix whose columns are the n eigenvectors \mathbf{e} of the eigenproblem “ $A \cdot \mathbf{e} = H \cdot \mathbf{e}$ ”, and let Λ be the corresponding diagonal matrix of n eigenvalues λ , so that $A \cdot E = H \cdot E \cdot \Lambda$. E^{-1} exists because the eigenvectors are linearly independent; and we can normalize them so that $E^T \cdot H \cdot E = I$. This last equation is not obvious; to simplify its proof assume temporarily that the n eigenvalues λ are distinct. Then the equation $E^T \cdot H \cdot E \cdot \Lambda = E^T \cdot A \cdot E \cdot \Lambda = (E^T \cdot A \cdot E)^T \cdot \Lambda = \Lambda \cdot E^T \cdot H \cdot E$ implies that $E^T \cdot H \cdot E$ is diagonal. We make $E^T \cdot H \cdot E = I$ by scaling the columns of E ; then $E^T \cdot A \cdot E = \Lambda$ exhibits eigenvalues in the same order as the columns of E exhibit eigenvectors.

Conversely, if $E^T \cdot H \cdot E = I$ and $\Lambda := E^T \cdot A \cdot E$ is diagonal for a square E then $A \cdot E = H \cdot E \cdot \Lambda$, so E 's columns are a full set of eigenvectors regardless of whether eigenvalues in Λ are distinct. Consequently, the eigenproblem “ $A \cdot \mathbf{e} = H \cdot \mathbf{e}$ ” is best solved by computing a *Simultaneous Congruence* E that renders $E^T \cdot H \cdot E = I$ while simultaneously rendering $\Lambda := E^T \cdot A \cdot E$ diagonal. If the eigenvalues in Λ are sorted, these equations determine each column of E uniquely within a sign-reversal except for columns belonging to repeated eigenvalues.

This eigenproblem has three pathologies:

- If some eigenvalues in Λ are repeated, their columns in E must be partially indeterminate.
- If H is merely semidefinite, some eigenvalues in Λ and their columns in E may be infinite.
- If H and A share a nullspace, some eigenvalue(s) and all eigenvectors are partially arbitrary.

Rounding errors threaten to degrade results computed from data too nearly pathological. “Too nearly ...” must be gauged in terms of the floating-point arithmetic's precision. As engineering computations migrate onto relatively inexpensive and very fast graphics boards optimized for 4-byte wide arithmetic adequate for computer games, pathologies judged too rare to care about when arithmetic was predominantly 8-bytes wide come to pose palpable threats.

Data too near the first pathology, repeated eigenvalues, can lead to eigenvectors some of which are too nearly linearly dependent. This is very unlikely but, if it occurs, it can be detected and remedied by means illustrated in §8 below. More likely, should eigenvalues be repeated, some formulas for the computation of eigenvectors will be derailed when floating-point arithmetic turns $0/0$ into unpredictable roundoff/roundoff. The way this contingency is treated in §2 and then in §11's program `gnsymeig` renders (near-)repeated eigenvalues innocuous.

Data too near the second pathology, a semidefinite H , can lead to eigenvectors and eigenvalues with such huge magnitudes that their rounding errors spill over into the rest of the eigensystem's computation, spoiling its accuracy. This situation calls for iterative refinement.

Data too near the third pathology, a nonzero intersection of the nullspaces of A and H , can turn roundoff into unnecessarily huge eigenvectors too nearly linearly dependent, and thereby incur excessive inaccuracies whose detection and remedy require costly supererogatory computation, including iterative refinement. An example is scrutinized in §8. Although this pathology occurs extremely rarely, it can occur; and then it is most likely to be overlooked, with consequences currently incalculable.

§2: Formulas for a 2-by-2 Example

We rely heavily upon this example later. Let $A := \begin{bmatrix} v_1 & \\ & v_2 \end{bmatrix}$ and $H := \begin{bmatrix} 1 & \\ & 1 \end{bmatrix}$, and assume $-1 < v_1 < 1$

to ensure that H is positive definite. Later we shall also assume that v_1 and v_2 are relatively tiny. The characteristic polynomial of the eigenproblem is

$$L(\lambda) := \det(\lambda H - A) = (1 - \lambda)^2 - (v_1 + v_2 - 2\lambda) + v_1 v_2 - \lambda^2.$$

Its discriminant's nonnegative square root is

$$:= \left((1 - \lambda)^2 + (v_1 + v_2 - 2\lambda)^2 \right)^{1/2};$$

temporarily assume it strictly positive. Then the eigenproblem's eigenvalues λ_1 and λ_2 are the distinct values $\frac{1}{2}(v_1 + v_2 - 2\lambda \pm \text{sgn}(\lambda_1 - \lambda_2)) / (1 - \lambda^2)$. They straddle both v_1 and v_2 between them since $L(v_2) = -(v_2 - \lambda)^2 < 0$ and $L(v_1) = -(v_1 - \lambda)^2 < 0$. Moreover λ_1 and λ_2 can be so ordered that $(\lambda_1 - \lambda_2) / (v_1 - v_2) = 1 / (1 - \lambda^2)$; then the eigenproblem's eigenvectors are the columns of

$$E := \begin{bmatrix} \sqrt{1 - v_2} & \frac{2 - \lambda}{\sqrt{v_1 - 2}} \\ \frac{1}{\sqrt{1 - v_2}} & \sqrt{v_1 - 2} \end{bmatrix} / (\text{sgn}(\lambda_1 - \lambda_2)), \quad (\text{sgn}(x) := x/|x| = \pm 1)$$

and are normalized to satisfy $E^T \cdot H \cdot E = I$ and $E^T \cdot A \cdot E = \text{Diag}([\lambda_1, \lambda_2])$. Consequently $A \cdot E = H \cdot E \cdot \lambda$. These equations are not so simple as they seem; although MAPLE 11 confirmed them almost instantly, DERIVE 4.11 could confirm them only indirectly.

Worse, this formula for E , though derived from λ in the usual way, is Numerically Unstable.

This formula loses its accuracy to roundoff for otherwise innocuous data A and H , namely when eigenvalues are too nearly equal. Only when the example's $A = \lambda H$ does $\lambda = 0$, and then $\lambda_1 = \lambda_2 = 0$ and E becomes indeterminate; any 2-by-2 E can satisfy $A \cdot E = H \cdot E \cdot \lambda$ without satisfying $E^T \cdot H \cdot E = I$ nor $E^T \cdot A \cdot E = \lambda \cdot I$. Consequently, when $A = \lambda H$ too closely, the example's formulas above produce an E that may well satisfy $A \cdot E = H \cdot E \cdot \lambda$ but severely violate $E^T \cdot H \cdot E = I$ and $E^T \cdot A \cdot E = \lambda \cdot I$ because of roundoff in the formulas. Try them! Roundoff afflicts similarly many other procedures that purport to solve the eigenproblem " $A \cdot e = H \cdot e \cdot \lambda$ ". For instance, MATLAB's $[E, \lambda] = \text{eig}(A, H)$ used to satisfy $A \cdot E = H \cdot E \cdot \lambda$ very closely, though $E^T \cdot H \cdot E$ and $E^T \cdot A \cdot E$ could depart substantially from diagonal, until MATLAB 6+, which now malfunctions only at data deemed to "deserve" it for H too nearly singular (λ too near ± 1).

The simplest presentation of better formulas for E uses trigonometric expressions starting with $\theta := \arcsin(\lambda)$ and $\phi := \arctan((2 - \lambda - (v_1 + v_2)) / ((v_1 - v_2) \cdot \cos(\theta)))$ between $\pm \pi/2$. Then

$$E := E(\lambda, \phi) := \begin{bmatrix} \cos \frac{\theta + \phi}{2} & -\sin \frac{\theta + \phi}{2} \\ \sin \frac{\theta - \phi}{2} & \cos \frac{\theta - \phi}{2} \end{bmatrix} / \cos(\theta) = \begin{bmatrix} \cos \frac{\theta}{2} & -\sin \frac{\theta}{2} \\ -\sin \frac{\theta}{2} & \cos \frac{\theta}{2} \end{bmatrix} \begin{bmatrix} \cos \frac{\phi}{2} & -\sin \frac{\phi}{2} \\ \sin \frac{\phi}{2} & \cos \frac{\phi}{2} \end{bmatrix} / \cos(\theta)$$

satisfies $E^T \cdot H \cdot E = I$ and $E^T \cdot A \cdot E = \text{Diag}([\lambda_1, \lambda_2])$ within roundoff no matter what angle between $\pm \pi/2$ replaces ϕ if $\arctan(0/0)$ is encountered, in which case reset $\phi := 0$ to minimize $\|E - I\|_F^2 := \text{Trace}((E - I)^T (E - I))$. Again, E has a positive diagonal and can be shown to so order that $(\lambda_1 - \lambda_2) / (v_1 - v_2) = 1 / (1 - \lambda^2) > 1$ if $v_1 < v_2$. These properties will be exploited later.

If the eigenvalues in λ differ too widely in magnitude the smaller may be obscured too badly by roundoff that is attenuated somewhat when the bigger eigenvalue, call it $\bar{\lambda}$, is used to recompute the smaller eigenvalue $\underline{\lambda}$ from the formula $\underline{\lambda} := (v_1 \cdot v_2 - \bar{\lambda}^2) / (\bar{\lambda} \cdot \cos(\theta))$.

Equivalent to the trigonometric formulas for E above are faster but more complicated purely algebraic formulas that could be used instead to compute E :

$$\begin{aligned} \text{If } |\theta| < 0.75 \text{ then } \cos(\theta) &:= (1 - \bar{\lambda}^2) \text{ else } \cos(\theta) := \sqrt{(1 - \bar{\lambda}^2) \cdot (1 + \bar{\lambda}^2)}; \\ \cos(\theta/2) &:= \frac{1}{2} \cdot (\sqrt{1 - \bar{\lambda}^2} + \sqrt{1 + \bar{\lambda}^2}); \quad \sin(\theta/2) := \frac{1}{2} \cdot \bar{\lambda} / \cos(\theta/2); \\ &:= (2 \cdot \bar{\lambda} - (v_1 + v_2) \cdot \bar{\lambda}) / ((v_1 - v_2) \cdot \cos(\theta)) = \tan(\theta); \quad \text{if } \bar{\lambda} \text{ is indeterminate reset } \bar{\lambda} := 0; \\ \beta &:= \frac{1}{2} / (1 + \bar{\lambda}^2) = \frac{1}{2} \cdot \cos(\theta); \quad \cos(\theta/2) := \sqrt{\beta + 1/2}; \quad \sin(\theta/2) := \beta \cdot \bar{\lambda} / \cos(\theta/2); \text{ etc.} \end{aligned}$$

The foregoing formulas' E \rightarrow I as A and H approach diagonal matrices, *i.e.* as $\bar{\lambda} \rightarrow 0$ and $\bar{\lambda} / (v_1 - v_2) \rightarrow 0$, except possibly if eigenvalues in λ coincide. This behavior will figure later in a criterion for stopping iterative refinement of nearly diagonal n -by- n matrices A and H .

On the other hand, when H is too nearly singular (when $|\bar{\lambda}|$ is too near 1), the foregoing formulas for E and θ suffer obscuration by roundoff amplified by a factor like $1/(1 - |\bar{\lambda}|)$. Some such amplification is unavoidable because $E \cdot E^T = H^{-1}$ so E has the biggest-singular-value norm $\|E\| = (\|H^{-1}\|) = 1/\sqrt{1 - \bar{\lambda}^2}$. Since at least one column of E must become almost as big as $\|E\|$, terms almost as big as $1/(1 - |\bar{\lambda}|)$ must appear during the evaluation of the matrix product $E^T \cdot H \cdot E$ before cancellation boils it down to I contaminated by amplified rounding errors. Also, at least one eigenvalue λ must become huge unless A is very nearly a scalar multiple of H , in which case E and an eigenvalue in λ may become practically indeterminate as all coefficients of $L(\lambda)$ cancel. Consequently bad things can happen numerically when $\bar{\lambda}$ comes too near ± 1 .

Only rarely is $|\sin(\theta)| = |\bar{\lambda}|$ so big, say $|\bar{\lambda}| > 0.75$, that E and θ are best computed from a formula that mitigates most of roundoff's ill effects by incurring critical cancellations before intermediate results are rounded off. Such a formula makes $E := X \cdot V^{-1} \cdot Y$ and $\theta := Y \cdot T \cdot Y$ for

$$Y := \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ \sin(\theta) & -\cos(\theta) \end{bmatrix}, \quad V := \begin{bmatrix} \sqrt{1 + |\bar{\lambda}|} & 0 \\ 0 & \sqrt{1 - |\bar{\lambda}|} \end{bmatrix} \cdot \bar{\lambda} \quad \text{and} \quad X := \begin{bmatrix} 1 & \text{sgn}(\bar{\lambda}) \\ \text{sgn}(\bar{\lambda}) & -1 \end{bmatrix}$$

in which sgn is the *signum* function (MATLAB's `sign`), so $\text{sgn}(\lambda) = \lambda/|\lambda|$ except $\text{sgn}(0) := 0$. To compute θ start with $T := V^{-1} \cdot (X \cdot (A \cdot X)) \cdot V^{-1}$ and set $\theta := \arctan(2 \cdot t_{12} / (t_{11} - t_{22})) / 2$; **DON'T DISREGARD PARENTHESES NOR SUBSTITUTE t_{21} FOR t_{12}** , which gets computed more accurately than t_{21} in critical cases. Then $\theta := \theta + (\text{sgn}(\bar{\lambda}) \cdot \text{sgn}(\theta) - 1) \cdot \text{sgn}(\bar{\lambda}) \cdot \pi / 4$. The use of $\text{sgn}(\dots)$ is intended to make this formula for E match the previous ones except for roundoff and the indeterminate case when $\bar{\lambda}$ encounters $0/0$, in which case resetting $\bar{\lambda} := 0$ is good enough.

Thus, if computed accurately enough, the columns of E are the normalized ($E^T \cdot H \cdot E = I$) eigenvectors of the example's 2-by-2 eigenproblem, and diagonal $\lambda = E^T \cdot A \cdot E$ exhibits the corresponding eigenvalues except for roundoff's interference, severe when $\|E\|$ is huge.

§3: Two Numerical 2-by-2 Examples

The following tests were carried out upon 4-byte data in 4-byte floating-point carrying 24 sig. bits. Three formulas above were tested: The first computes eigenvalues in λ first and then, from them, eigenvectors in E . The second formula computes E first and then, from it, λ . The third formula, intended to be used only when H is nearly singular, first applies a congruence X to induce cancellation before roundoff and then applies congruences V^{-1} and Y to reveal E and λ . Strictly algebraic (using only $+$, $-$, \cdot , $/$ and $\sqrt{\quad}$, no trigonometry) versions of the second and third formulas were used to insulate results from the vagaries of math. libraries, though the best libraries nowadays produce slightly better results from the trigonometric versions. Enough digits are displayed to let decimal-binary conversion reproduce all binary data and results perfectly.

Displayed also are *Normalized Residuals* to indicate how well or poorly computed results satisfy the equations $E^T \cdot H \cdot E = I$, $E^T \cdot A \cdot E = \lambda$ and $A \cdot E = H \cdot E$ compared with the uncertainty these equations accrete when their constituents are computed in 24-sig.-bit arithmetic whose roundoff bound here is $\epsilon/2 := 2^{-24}$. Residual eHe is the biggest element of $|E^T \cdot H \cdot E - I| ./ (|E^T| \cdot |H| \cdot |E|)$ in which the absolute values $|\dots|$ and division $./$ apply *elementwise* to their respective arrays, but \cdot is matrix multiplication as usual. Similarly, residual eAe is the biggest element of $|E^T \cdot A \cdot E - \lambda| ./ (|E^T| \cdot |A| \cdot |E|)$, and the biggest element of $|A \cdot E - H \cdot E| ./ (|A| \cdot |E| + |H| \cdot |E|)$ is $eAeH$. A residual so normalized gets much bigger than 3 only when its equation is dissatisfied much worse by computed results E and λ than by the ideal E and λ and data A and H after all four have been muddied by roundoff-like perturbations in their last digits stored.

1st Data: $v_1 := 13981013/2^{23}$; $v_2 := 13981011/2^{23}$; $\lambda := 13981012/2^{26}$; $\lambda := (2^{23} - 125)/2^{26}$

$$A = \begin{bmatrix} 1.666666627 & 0.208333313 \\ 0.208333313 & 1.666666388 \end{bmatrix} \quad H = \begin{bmatrix} 1 & 0.124998137 \\ 0.124998137 & 1 \end{bmatrix}$$

Accurate $E = \begin{bmatrix} 0.680938538 & -0.743098407 \\ 0.652154220 & 0.768483837 \end{bmatrix}$ $\text{diag}(\lambda) = \begin{bmatrix} 1.666669269 \\ 1.666662958 \end{bmatrix}$

Note that eigenvalues are nearly equal.

1st Formula's $E = \begin{bmatrix} 0.664820254 & -0.740706980 \\ 0.672046542 & 0.771829128 \end{bmatrix}$ $\text{diag}(\lambda) = \begin{bmatrix} 1.666669130 \\ 1.666662931 \end{bmatrix}$

Normalized Residuals $eHe = 2.08e5$ $eAe = 2.08e5$ $eAeH = 0.68$

Unacceptable eigenvectors produced for near-equal eigenvalues.

2nd Formula's $E = \begin{bmatrix} 0.680961311 & -0.743077636 \\ 0.652130783 & 0.768503785 \end{bmatrix}$ $\text{diag}(\lambda) = \begin{bmatrix} 1.666669607 \\ 1.666663170 \end{bmatrix}$

Normalized Residuals $eHe = 1.41$ $eAe = 0.35$ $eAeH = 0.85$

Acceptable eigenvectors considering their hypersensitivity to roundoff.

3rd Formula's $E = \begin{bmatrix} 0.680401206 & -0.743590474 \\ 0.652709603 & 0.768012285 \end{bmatrix}$ $\text{diag}(\lambda) = \begin{bmatrix} 1.666669369 \\ 1.666662812 \end{bmatrix}$

Normalized Residuals $eHe = 0.89$ $eAe = 1.48$ $eAeH = 0.29$

Eigenvectors somewhat worse than the second formula's.

2nd Data: $v_1 := v_2 := 13981013/2^{23}$; $:= 13981012/2^{23}$; $:= 1 - 5/2^{24}$

$$A = \begin{bmatrix} 1.666666627 & 1.666666508 \\ 1.666666508 & 1.666666627 \end{bmatrix} \quad H = \begin{bmatrix} 1 & 0.999999702 \\ 0.999999702 & 1 \end{bmatrix}$$

Accurate $E = \begin{bmatrix} 1295.268930 & 0.500000037 \\ -1295.268930 & 0.500000037 \end{bmatrix}$ $\text{diag}(\) = \begin{bmatrix} 0.400000000 \\ 1.666666816 \end{bmatrix}$

Note the huge eigenvector and that H is very nearly singular.

1st Formula's $E = \begin{bmatrix} 1330.762329 & 305.2990417 \\ -1330.762329 & 305.2973938 \end{bmatrix}$ $\text{diag}(\) = \begin{bmatrix} 0.400000036 \\ 1.600000143 \end{bmatrix}$

Normalized Residuals $eHe = 8.39e6$ $eAe = 8.39e6$ $aeher = 1.71e5$

Unacceptable results produced because H is too nearly singular.

2nd Formula's $E = \begin{bmatrix} 1295.268921 & 0.499896228 \\ -1295.268921 & 0.499896228 \end{bmatrix}$ $\text{diag}(\) = \begin{bmatrix} 0.632455528 \\ 1.665974736 \end{bmatrix}$

Normalized Residuals $eHe = 3.48e3$ $eAe = 0.46$ $aeher = 1.74e3$

Unacceptable results produced because H is too nearly singular.

3rd Formula's $E = \begin{bmatrix} 0.500000000 & -1295.268921 \\ 0.500000000 & 1295.268921 \end{bmatrix}$ $\text{diag}(\) = \begin{bmatrix} 1.666666746 \\ 0.400000006 \end{bmatrix}$

Normalized Residuals $eHe = 1.25$ $eAe = 0.90$ $aeher = 0.18$

Exemplary results despite that H is nearly singular.

The foregoing results corroborate the error-analyses that preceded them:

- When eigenvalues are so nearly coincident that eigenvectors are partially indeterminate, the first formula's eigenvectors in E can be very wrong.
- When H is so nearly singular that at least one eigenvector in E is enormous, only the third formula's results can be expected to produce tolerable normalized residuals.

But the foregoing results are utterly unrealistic. They rely upon too many unlikely assumptions:

- An error-analysis has been performed. Error-analyses are attempted only rarely.
- Accurate results are known for comparison. Accurate results are knowable only rarely.
- Residuals were computed to help assess accuracy. Residuals get computed only rarely.
- Three numerical methods were compared. More than one are available only rarely.

What can be done when a numerical program is problematical because its results' consequences (perhaps remote) for one data-set have aroused suspicion? What can be done to assuage, deflect or focus suspicion? If part of the program deserves closer scrutiny, how can it be localized?

By using techniques described in §14 of my web page's <.../Mindless.pdf>, a formula above for E and H has been excised from a lengthy program whose results aroused suspicion one day. Suppose one of that lengthy program's big data-sets produced intermediate results from which, after arduous labor, one of the two numerical 4-tuples called "Data" above was extracted. How did this 4-tuple attract attention to itself? Its four results from the formula in question dispersed too widely when re-evaluated with arithmetic's rounding altered in each of four ways:

- Rounded to Nearest (as usual)
- Rounded Down toward $-$
- Roundes Up toward $+$
- Rounded toward Zero

The results are displayed below with x 's in place of divagated digits:

1st Data: $v_1 := 13981013/2^{23}$; $v_2 := 13981011/2^{23}$; $:= 13981012/2^{26}$; $:= (2^{23} - 125)/2^{26}$

$$A = \begin{bmatrix} 1.666666627 & 0.208333313 \\ 0.208333313 & 1.666666388 \end{bmatrix} \quad H = \begin{bmatrix} 1 & 0.124998137 \\ 0.124998137 & 1 \end{bmatrix}$$

1st Formula's $E = \begin{bmatrix} 0.7x & -0.7x \\ 0.7x & 0.8x \end{bmatrix}$ $\text{diag}(\) = \begin{bmatrix} 1.666669x \\ 1.666663x \end{bmatrix}$
 Unacceptable eigenvectors produced for near-equal eigenvalues.

2nd Formula's $E = \begin{bmatrix} 0.6809x & -0.7431x \\ 0.652x & 0.768x \end{bmatrix}$ $\text{diag}(\) = \begin{bmatrix} 1.66667x \\ 1.66666x \end{bmatrix}$
 Acceptable eigenvectors considering their hypersensitivity to roundoff.

3rd Formula's $E = \begin{bmatrix} 0.680x & -0.743x \\ 0.652x & 0.768x \end{bmatrix}$ $\text{diag}(\) = \begin{bmatrix} 1.666669x \\ 1.66666x \end{bmatrix}$
 Eigenvectors somewhat worse than the second formula's.

2nd Data: $v_1 := v_2 := 13981013/2^{23}$; $:= 13981012/2^{23}$; $:= 1 - 5/2^{24}$

$$A = \begin{bmatrix} 1.666666627 & 1.666666508 \\ 1.666666508 & 1.666666627 \end{bmatrix} \quad H = \begin{bmatrix} 1 & 0.999999702 \\ 0.999999702 & 1 \end{bmatrix}$$

1st Formula's $E = \begin{bmatrix} 1xxx.x & \pm xxx.x \\ -1xxx.x & xxx.x \end{bmatrix}$ $\text{diag}(\) = \begin{bmatrix} 0.4000000x \\ 2.x \end{bmatrix}$
 Unacceptable results produced because H is too nearly singular.

2nd Formula's $E = \begin{bmatrix} 1295.269x & 0.500x \\ -1295.269x & 0.500x \end{bmatrix}$ $\text{diag}(\) = \begin{bmatrix} 0.x \\ 1.66x \end{bmatrix}$
 Unacceptable results produced because H is too nearly singular.

3rd Formula's $E = \begin{bmatrix} 1295.269x & 0.5000000x \\ -1295.269x & 0.5000000x \end{bmatrix}$ $\text{diag}(\) = \begin{bmatrix} 0.4000000x \\ 1.666667x \end{bmatrix}$
 Exemplary results despite that H is nearly singular.

Thus, recomputation with redirected roundings has helped focus suspicion upon a short stretch of a long program using the first or second formula. *Suspicion* is not *conviction*; no formula can be condemned until after an error-analysis without which, for all we know, the formula might have been designed to work correctly only when arithmetic is rounded to Nearest as usual. Some math libraries' divide, sqrt, elementary transcendental functions and Binary Decimal conversions are like that. Neither can a formula be exonerated until after an error-analysis without which, for all we know, the formula might deliver very nearly the same wrong result regardless of roundoff's redirection, though such a formula is probably (not surely) wrong algebraically absent roundoff.

Think of recomputation with redirected roundings as a way to prioritize the search for aberrant constituents of a numerical program. We put subprograms or formulas that seem excessively sensitive to roundoff at the head of the list of candidates for more intensive scrutiny.

Results from the foregoing sets of data and others, many generated randomly, support these ...

§4: Conclusions Drawn from an Error-Analysis of the 2-by-2 Example:

- A relatively small residual $R := A \cdot E - H \cdot E$ testifies to the accuracy of E but maybe not E .
- All three residuals R , $E^T \cdot H \cdot E - I$ and $E^T \cdot A \cdot E - V$ are relatively small when E is accurate, but also if it has lost accuracy because eigenvalues nearly coincide or H is nearly singular.
- Eigenvectors E from the first formula above can be quite wrong if eigenvalues are too close.
- Accuracies of the first and second formulas above deteriorate when H is too nearly singular.
- The third formula's accuracy tends to improve as H becomes more nearly singular.

Consonant with these conclusions is a policy that employs the second formula when $| \lambda_i - \lambda_j | < 3/4$, and the third formula when $3/4 < | \lambda_i - \lambda_j | < 1$, to get results E and V at least about as accurate as the data's and arithmetic's precisions deserve in so far as all three of these results' residuals are relatively tiny. (Cf. §8.) This policy's computational cost barely exceeds the cost of a policy that would employ any one of those three formulas above for all data.

§5: Iterative Refinement of All n Eigenvectors

Now let F be any matrix that approximates the desired matrix E of n eigenvectors well enough that $F^T \cdot H \cdot F$ and $F^T \cdot A \cdot F$ are nearly diagonal. Moreover we normalize the columns of F to make $F^T \cdot H \cdot F = \text{Diag}(F^T \cdot H \cdot F) = I$ and $F^T \cdot A \cdot F = V := \text{Diag}(F^T \cdot A \cdot F)$ so that the n diagonal elements of V exhibit our currently best estimates of the desired eigenvalues. To refine the accuracy of F we shall generate a sequence of improved approximations F_1, F_2, F_3, \dots converging rapidly to E by using a process analogous to Jacobi's iteration for the eigensystem of a symmetric matrix:

Starting from $F_0 := F$, each $F_{k+1} := F_k \cdot E_k$ wherein E_k differs from the n -by- n identity I in the four elements of a 2-by-2 principal submatrix. This submatrix of E_k is the normalized eigenvector matrix of the 2-by-2 example's eigenproblem taken from the corresponding 2-by-2 principal submatrices of $A_k := F_k^T \cdot A \cdot F_k$ in place of the example's A , and of $H_k := F_k^T \cdot H \cdot F_k$ in place of its H . In the locations where E_k has nonzero off-diagonal elements, corresponding off-diagonal elements of A_{k+1} and H_{k+1} are annihilated, while $\text{Diag}(H_{k+1}) = I$ is preserved. But doing so puts small nonzero elements back into some locations where A_k and H_k had zeros.

As iteration converges, $A_k \rightarrow A$ and $H_k \rightarrow H$; and $E_k \rightarrow E$ since $\text{diag}(E_k) > \mathbf{0}$. Convergence is obviously quadratic if all eigenvalues are distinct and $F_0^T \cdot H \cdot F_0$ and $F_0^T \cdot A \cdot F_0$ start near enough to diagonal. The iteration always converges regardless of the initial normalized F ; this is proved in §6. The speed of convergence is a very interesting question whose answer seems to depend upon how some further questions are answered. ...

In what order should off-diagonal elements be annihilated? The first, if not the best, order that comes to mind is *Lexicographic*: (1, 2), (1, 3), (1, 4), ..., (1, n), (2, 3), (2, 4), ..., (2, n), (3, 4), (3, 5), ..., ..., ($n-2$, $n-1$), ($n-2$, n), ($n-1$, n) constitutes one pass over all off-diagonal elements.

Repeated passes drive all off-diagonal elements towards zero provided the diagonal elements of A_k are kept sorted. Whether keeping them sorted is necessary or advantageous for convergence is an interesting question for research. Another is whether an order better than Lexicographic exists. These questions bear upon the practicality of iterative refinement of sparse or banded systems of huge dimensions. Currently only systems of modest dimensions get refined quickly by the iteration's experimental MATLAB implementation in §11. To refine huge systems quickly will require the incorporation of perhaps tricky task-switching to perform anticipatory fetches from DRAM to fill cache lines just in time for arithmetic consumption. To keep concurrent processors busy will require programs like those in §§8.4.6-7 of Golub & Van Loan (1996).

Regardless of the order chosen for annihilation, matrices E_k that would annihilate elements of A_k and H_k already tinier than some suitably declining threshold should be skipped to save time. What is a "suitable" sequence of thresholds? They cannot apply solely to differences $\|E_k - I\|_F$ since some of these can be almost arbitrarily big for arbitrarily tiny off-diagonal elements when eigenvalues are repeated. Suitable thresholds may depend upon the order chosen for off-diagonal annihilations. If the order is Lexicographic, for instance, an off-diagonal element already rather tinier than average in its row and column might not be worth annihilating yet; and a row whose elements are all too much tinier than other rows' averages might not yet be worth scanning for elements worth annihilating. "Average" implies that a running sum of each row's off-diagonal magnitudes or their squares must be updated after each annihilation, which is feasible though it adds noticeably to the cost of each annihilation. This cost, significantly greater than the cost of simpler threshold schemes explored by D. Corneil (1965), J.H. Wilkinson (1965) and others mentioned in ch. 9 of Parlett (1998) and ch. 8.4 of Golub & Van Loan (1996), was incurred to cope with occasional concentrations of huge elements in a few rows and columns of some A_k after their growth revealed H to be nearly singular. Choices of "suitable" thresholds and how best to exploit them remain questions for research even though some rather different choices have worked about equally well.

The iteration is incremental. Each application of an E_k alters two columns of $F_{k+1} := F_k \cdot E_k$ and two rows and two columns of $A_{k+1} := E_k^T \cdot A_k \cdot E_k$ and of $H_{k+1} := E_k^T \cdot H_k \cdot E_k$, so one pass over all off-diagonal elements costs time proportional to n^3 . Quadratic convergence entails so few passes that adequate accuracy for all eigenvectors should cost time proportional to n^3 . Whether this cost estimate is valid in all cases is an interesting question for research. In particular, ...

Test matrices of the contrived form $A = \begin{bmatrix} 2X & X \\ X & X \end{bmatrix}$ and $H = \begin{bmatrix} 2Y & Y \\ Y & Y \end{bmatrix}$ have every eigenvalue repeated.

Consequently the iteration described above converges slowly at first, the more so if the matrices' dimension is large or H is nearly singular, and yet the iteration terminates after few if any more annihilations than if eigenvalues were well separated. Why?

Another question for research is ...

- What should be done if A and H are complex Hermitian instead of real symmetric?

§6: Why Does the Iteration Converge?

For about forty years structural engineers have run iterations like the one described above, whose 2-by-2 congruences E_k annihilate two off-diagonal elements of $A_{k+1} := E_k^T \cdot A_k \cdot E_k$ and two of $H_{k+1} := E_k^T \cdot H_k \cdot E_k$, to find vibrational modes and frequencies of undamped elastic structures, but with no proof that iteration always converges. Ch. 15 of Parlett's (1998) book has accounts and citations. I believe the first proof that such an iteration's sequence $F_{k+1} := F_k \cdot E_k$ converges from every invertible F_0 is outlined hereunder. The outline begins with a nearly obvious ...

Lemma: If h is the biggest magnitude of the off-diagonal elements of a positive definite n -by- n matrix H whose diagonal $\text{Diag}(H) = I$, then $0 < \det(H) \leq 1 - h^2$. And if also $0 < h \cdot (n-1) < 1$ then $\det(H) > (1 - (n-1) \cdot h)^n$.

Proof: Since $\det(H) = \det(P \cdot H \cdot P^T)$ for every permutation matrix P , choose it to put a biggest off-diagonal element $\pm h$ into the first row and second column. The *Choleski* factorization $P \cdot H \cdot P^T = U^T \cdot U$ exhibits $P \cdot H \cdot P^T$ as a product of a real upper-triangular matrix U and its transpose U^T . Each diagonal element u_{ii} of $P \cdot H \cdot P^T$ is the sum of the squares of all elements in a corresponding column of U , so none of its elements can exceed 1 in magnitude. This holds particularly for U 's diagonal elements, whose product is $\det(U)$; and the second diagonal element is $(1 - h^2)$. Therefore $\det(H) = \det(U)^2 \leq 1 - h^2$ as claimed.

Next suppose $0 < h \cdot (n-1) < 1$, and set n -by-1 column $u := [1, 1, \dots, 1]^T$. Then in magnitude $|H - I| \leq h \cdot (u \cdot u^T - I)$ elementwise. Apply *Gershgorin's Circle Theorem* to infer that all n eigenvalues of H lie between $1 \pm (n-1) \cdot h$. The eigenvalues' product is $\det(H)$, so it cannot be less than $(1 - (n-1) \cdot h)^n$, and it cannot be that small since the eigenvalues' sum is $\text{Trace}(H) = n$. Thus the Lemma's lower bound for $\det(H)$ is proved. It will be adequate for our purposes though much smaller than a lower bound $(1 - (n-1)^2 \cdot h^2)^{n/2}$ that takes rather longer to prove.

Each iteration $H_{k+1} := E_k^T \cdot H_k \cdot E_k$ annihilates some off-diagonal element — call it h_k — of H_k while preserving $\text{Diag}(H_{k+1}) = \text{Diag}(H_k) = I$. The congruence E_k has $\det(E_k) = 1 / (1 - h_k^2)$ according to all the formulas for E in §2, so $\det(H_{k+1}) = \det(H_k) / (1 - h_k^2)$. Since the iteration visits every super-diagonal location infinitely often to annihilate any bigger-than-average element of A_k or H_k found there, the sequence $\{\det(H_k)\}$ is monotone non-decreasing and converges up to a limit; call it $1 - \bar{h}^2$ where $0 < \bar{h} < 1$. Later we shall find $\bar{h} = 0$, but now we assume for the sake of an argument by contradiction that $\bar{h} > 0$. Let \bar{h} be the greatest lower bound of the biggest magnitudes of off-diagonal elements of all positive definite n -by- n matrices H with $\text{Diag}(H) = I$ and $\det(H) = 1 - \bar{h}^2$; the Lemma implies that $\bar{h} > 0$ if $\bar{h} > 0$.

The convergence of $\{\det(H_k)\}$ to a positive limit implies that $\det(E_k) \rightarrow 1$, which implies that the previous paragraph's $h_k \rightarrow 0$ despite that every H_k had at least two elements no smaller than \bar{h} in magnitude. Therefore all but finitely many congruences $E_k = E(\theta_k, \phi_k)$ are determined by the second formula, not the third in §2, and their $\theta_k \rightarrow 0$ so that all but finitely many of the congruences E_k come arbitrarily close to rotations $E(\theta_k, 0)$ through angles θ_k between $\pm \pi/4$ calculated to annihilate bigger-than-average off-diagonal elements of A_k regardless of h_k .

Were $\bar{h} > 0$, the rotations would somehow circulate the biggest elements of H_k , at least two no smaller than \bar{h} , to evade annihilation. Our iteration would ultimately approximate arbitrarily closely a classical Jacobi iteration whose rotations diagonalize any given real symmetric matrix. Its rotations are known to converge to I quadratically even if eigenvalues are repeated; see ch. 9 of Parlett (1998) or ch. 8.4 of Golub & Van Loan (1996) and the references they cite. But then the matrices H_k would converge to a limit with at least two off-diagonal elements no smaller in magnitude than \bar{h} and eligible for annihilation because they are bigger than average. This would contradict the convergence of $\det(E_k)$ to 1. Instead $\bar{h} = 0$, so $H_k \rightarrow I$ and A_k quadratically, as claimed. End of Proof's Outline

This proof is a lucky consequence of a decision to force every $\text{Diag}(H_k) = \text{Diag}(H_0) = I$ by diagonal scaling instead of leaving the diagonals unconstrained. Leaving them unconstrained would not change the eigenvalues: If the 2-by-2 congruences E_k drove $H_k \rightarrow Y^2 = \bar{E}^T \cdot H \cdot \bar{E}$ and $A_k \rightarrow W = \bar{E}^T \cdot A \cdot \bar{E}$, both limits being diagonal, then the eigenvalues in $\lambda = W \cdot Y^{-2}$ would be the same as before except for their order, and the reordered eigenvectors in $\bar{E} = E \cdot Y$ would be the same as before except for column-scaling. But different criteria for choosing elements big enough to be worth annihilating would alter the order of annihilations of off-diagonal elements of A_k and H_k . Can “Big” be distinguished from “Small” in a way independent of diagonal scaling?

“Big” and “Small” should be gauged relative to the size of the data and its nearness to whatever pathologies may undermine accuracy. Singular matrices H would cause the worst damage. These form a cone through O in the space of n -by- n symmetric matrices. One plausible measure of the nearness of the given matrix H to that cone is the angle between them subtended at O ; this angle is $\arcsin(1/\kappa(H))$ wherein $\kappa(H) := \|H\| \cdot \|H^{-1}\|$ is the *Condition Number* of H , a frequently cited measure of the sensitivity of H^{-1} to perturbations of H . But $\kappa(H)$ can differ drastically from $\kappa(D \cdot H \cdot D)$ if the elements of the diagonal matrix D vary too wildly, whereas changing data from $\{A, H\}$ to $\{D \cdot A \cdot D, D \cdot H \cdot D\}$ changes eigenvalues in λ not at all, and changes eigenvectors in E trivially to $D^{-1} \cdot E$. Perhaps $\kappa(H)$ is not so appropriate a measure of the sensitivity of H^{-1} to perturbations of H *that matter* as its frequency of citation suggests. For further exploration of this issue see my web page's [.../Math128/FailMode.pdf](http://math128/failmode.pdf).

To immunize the congruences E_k against an accident of diagonal scaling, one must be applied initially so that given data $\{D \cdot A \cdot D, D \cdot H \cdot D\}$ will get scaled back to distinguished data $\{A, H\}$ independent of D . The diagonal scaling that makes $\text{Diag}(H_0) = I$ was chosen because of a theorem of A. van der Sluis (1969) that says then $\kappa(H_0)$ cannot exceed $n \cdot \min_D \kappa(D \cdot H \cdot D)$. Besides, this roughly minimized condition number has never been enormously big for any mechanical vibration problem that I have seen. These thoughts motivated forcing $\text{Diag}(H_k) = I$; the consequent proof of convergence was a serendipitous byproduct.

§7: Extra-Precise Accumulation of Residuals

The obvious residual $A \cdot F - H \cdot F \cdot V$ reveals too little. Better, after F has been scaled to make $\text{Diag}(F^T \cdot H \cdot F) = I$, is to compute two residuals $V := F^T \cdot A \cdot F - V$ and $I := F^T \cdot H \cdot F - I$ each with $\text{diag}(V) = \text{diag}(I) = \mathbf{0}$. If the cost is tolerable, these residuals should be accumulated extra-precisely before cancellation; all additions/subtractions should be extra-precise, and then multiplications will involve at most one extra-precise factor. Then apply iterative refinement to annihilate the off-diagonal elements of $V + V$ and $I + I$. Then the algebraic formulas for the 2-by-2 matrices E_k should be altered to compute each small $E_k := E_k - I$ relatively accurately, and likewise alter $F_{k+1} := F_k \cdot E_k = F_k + F_k \cdot E_k$ to accumulate a lengthy sequence of successive matrices F_k a little extra-precisely with the aid of *Compensated Summation* described in ch. 4 of N.J. Higham's (2002) book. The same goes for

$$I_{k+1} := H_{k+1} - I = I_k + (E_k^T \cdot V_k + (E_k^T \cdot I_k)^T + (E_k^T \cdot H_k) \cdot E_k)$$

and similarly for V_k and $V_{k+1} - V_k$; the intricate details are a story for another day.

Especially when some roughly computed eigenvalues in V and their normalized eigenvectors among the columns of F are rather bigger than the others, most of the benefit of extra-precise accumulation of matrix products accrues during the very first matrix multiplications that produce $A_0 := F^T \cdot (A \cdot F)$ and $H_0 := F^T \cdot (H \cdot F)$ for subsequent refinement with none of the intricacies of the previous paragraph. Old versions of MATLAB running on old 68040-based Macintoshes and Wintel PCs accumulated matrix products by default in registers with 64 sig. bits before storing each accumulation's leading 53 sig. bits into the matrix product in memory. That way was the most convenient way to use the hardware. Then old versions of the MATLAB program `gnsymeig` listed in §11, invoked to refine initial results obtained from $[F, V] = \text{eig}(A, H)$, routinely improved the less accurate among these initial results by two or three sig. dec., often far more.

Extra-precise arithmetic is available in the hardware of most computers on which MATLAB runs, but inaccessible at a bearable cost. Consequently `gnsymeig`, an experimental MATLAB program in §11 that implements the Jacobi-like iteration discussed above, does not accumulate residuals extra-precisely. It was motivated by MATLAB 5's results from $[F, V] = \text{eig}(A, H)$, which can be complex if H is too nearly semi-definite, and otherwise $F' \cdot A \cdot F$ and $F' \cdot H \cdot F$ can be far from diagonal. The program $[E, v] = \text{gnsymeig}(F' \cdot A \cdot F, F' \cdot H \cdot F)$ has cleaned up MATLAB 5's results by overwriting real $F = F \cdot E$ (or better $F = F + F \cdot (E - I)$ if $E \neq I$) and then replacing $\text{diag}(V)$ by v . Similarly cleaned up results from MATLAB 6.5 have improved substantially, mostly when dimensions were big and/or H was not far from singular, provided the invocation

```
system_dependent('setprecision', 64)
```

on PCs preceded `gnsymeig(...)` to enhance the accuracy of some critical matrix multiplications like the initial $H \cdot E$ and $A \cdot E$. But that invocation seems ineffective in MATLAB 7.4, whose misbehavior I do not yet understand.

How does MATLAB 6.5 compute $[F, V] = \text{eig}(A, H)$? According to its documentation, after $U = \text{chol}(H)$ produces an upper-triangular factor of $H = U^T \cdot U$, the equation $U^T \cdot W \cdot U = A$ is solved for the symmetric W with less work than $W = U' \cdot A \cdot U$. Then $[Q, V] = \text{eig}(W)$ computes the sorted diagonal V of eigenvalues and an orthogonal $Q = Q^{-T}$ of eigenvectors, and then $F = U \cdot Q$. I have failed to confirm this description by experiments.

§8: How Well at Best Can `gnsymeig` Work?

Ideally, two numbers, *Speed* and *Accuracy*, should answer this question about the experimental program presented in §11. Actually, complications obstruct every approach to the ideal.

A hardware-independent assessment of speed can no longer be inferred from a count of floating-point arithmetic operations now that most of them, all but perhaps divisions and square roots, cost so much less time than out-of-cache memory accesses. So, instead of arithmetic operations, `gnsymeig` counts four kinds of operations:

- k_{steps} := how many Jacobi-like congruences E_k were executed each to annihilate two pairs of off-diagonal elements of A_k and H_k of which at least one pair was bigger than average. Each such congruence costs time proportional to n , so k_{steps} has been expected not to exceed a modest multiple (like 4) of n^2 .
- k_{sweeps} := how many *sweeps*, each visiting all $n(n-1)/2$ superdiagonal locations to see whether they hold any bigger-than-average off-diagonal elements, have been executed. Each such sweep costs time proportional to n^2 plus time spent on the sweep's annihilations counted separately by k_{steps} , so k_{sweeps} has been expected not to exceed a modest fraction (like 1/2) of n .
- k_{sorts} := the number of times when a sweep had to be preceded by a rearrangement of the rows and columns of A_k and H_k to impose or restore the increasing order of the diagonal elements of A_k . This was expected to occur infrequently, but it has occurred several times when many eigenvalues were nearly repeated, and each occasion cost time proportional to n^2 , so perhaps a cheaper but more complicated way (suggested by B.N. Parlett) to preserve order should be reconsidered.
- k_{bigH} := the number of congruences E_k determined by §2's third formula whenever $h_k \geq 3/4$. This cannot happen often unless initially H_0 is practically singular.

All four counts were expected to be relatively small for nearly diagonal initial data A and H of a kind `gnsymeig` was intended to handle well. But “the best laid schemes of mice and men ...”

The iteration's stopping criterion influences counts k_{steps} and k_{sweeps} , inflating them greatly if accuracy is sought beyond what the computer's arithmetic can achieve economically. Thus do we become entangled in complicated questions that beset assessments of accuracy:

- How should we measure (absolute? relative? ...?) accuracy?
- How much accuracy do the data deserve?
- How much accuracy can be achieved economically with the computing resources available?
- How close will a chosen program come to achieving whatever accuracy is deserved and/or achievable economically?

At first sight the accuracy deserved by the data is determinable from the computed results thus:

Suppose the given data $\{A, H\}$ is augmented by given arrays $\{\overline{A}, \overline{H}\}$ of tiny positive numbers representing the uncertainty tolerable in the given data in so far as A is deemed indistinguishable from $A + \overline{A}$ for practical purposes whenever $|A| \geq \overline{A}$ elementwise, and likewise for $H + \overline{H}$.

Normally $\{\bar{A}, \bar{H}\}$ are somewhat bigger than roundoff in $\{A, H\}$ but far tinier than the uncertainties that data inherit from the physical and geometrical uncertainties in parameters (like strength and length) that determine the elements of $\{A, H\}$. These are usually rather more numerous than the parameters whose physical and geometrical variations induce variations in $\{A, H\}$ that are correlated in ways mere numerical analysts cannot know. Therefore perturbations induced by the finite precisions of memory and arithmetic must be kept far tinier than these variations lest computational artifacts blight computed results beyond the bounds of physics or geometry. For some simpler geometrical examples of that kind of blight see §§11 & 17 of my web page's [.../MathH110/Cross.pdf](http://www.math.hawaii.edu/~kahan/Cross.pdf)

Perturbations A and H perturb eigenvalues in ways predictable with the aid of eigenvectors: If $A\mathbf{e} = \lambda\mathbf{e}$ and $\mathbf{e}^T H \mathbf{e} = 1$ then changing A to $A + \Delta A$ and H to $H + \Delta H$ changes λ to $\lambda + \Delta\lambda$ with $\Delta\lambda = \mathbf{e}^T (\Delta A - \lambda \Delta H) \mathbf{e}$ if terms $(\dots)^2$ are ignored. This simple approximation to λ obtained by differentiation holds for simple eigenvalues, and for repeated eigenvalues too provided \mathbf{e} is forced to range over all normalized ($\mathbf{e}^T H \mathbf{e} = 1$) eigenvectors belonging to λ .

Roughly, then, $|\Delta\lambda| \approx |\mathbf{e}^T (\Delta A + |\lambda| \Delta H) \mathbf{e}|$ elementwise, which reveals the uncertainty $\bar{\lambda}$ that an eigenvalue inherits from the uncertainties $\{\bar{A}, \bar{H}\}$ in the given data $\{A, H\}$.

This inheritance $\bar{\lambda}$ from data's uncertainty also indicates how much accuracy the data deserve in an eigenvalue estimate v : If its eigenvector estimate \mathbf{f} is accurate enough and normalized to make $\mathbf{f}^T H \mathbf{f} = 1$, we should not complain about an error $|\bar{\lambda} - v|$ not much bigger than

$$\bar{\lambda} = |\mathbf{e}^T (\bar{A} + |\lambda| \bar{H}) \mathbf{e}| \quad |\mathbf{f}^T (\bar{A} + |v| \bar{H}) \mathbf{f}| \text{ elementwise.}$$

Of course, this is not worth computing unless either ...

- λ is known and the accuracy of the program that computed v and \mathbf{f} is being tested, or else
- λ is unknown but the estimates v and \mathbf{f} are thought to be about as accurate as data deserve.

Even with no estimated eigenvalues nor eigenvectors the formula for $\bar{\lambda}$ exposes something well worth knowing:

At least one eigenvalue is threatened by extreme uncertainty whenever H is too nearly singular.

Why? The normalization $\mathbf{e}^T H \mathbf{e} = 1$ implies $\|\mathbf{e}\|^2 = \|H^{-1}\|$, which overestimate supplies an upper bound for uncertainty, namely

$\bar{\lambda} = |\mathbf{e}^T (\bar{A} + |\lambda| \bar{H}) \mathbf{e}| \leq \|\mathbf{e}\|^2 \cdot (\|\bar{A}\| + |\lambda| \|\bar{H}\|) = \|H^{-1}\| \cdot (\|\bar{A}\| + |\lambda| \|\bar{H}\|)$, that can exceed $\|\bar{A}\| + |\lambda| \|\bar{H}\|$ enormously whenever H is too nearly singular. This upper bound is not excessively pessimistic; the eigenvectors' normalization $E^T H E = I$ compels at least one eigenvector \mathbf{e} to have a huge $\|\mathbf{e}\|^2 = \|E\|^2/n = \|H^{-1}\|/n$. Therefore the threat is real.

Rarely is enough information available to compute $\bar{\lambda}$ and apply it, yet a notion like inherited uncertainty remains pertinent to an assessment of a program's accuracy. This notion concerns the uncertainty added to computed residuals $\mathbf{I} := F^T H F - I$ and $\mathbf{V} := F^T A F - V$ by roundoff's accumulation during their computation from the program's output $V = \text{Diag}(F^T A F)$ and F . If the residuals could be computed accurately enough they would provide rough estimates for the errors in computed eigenvalues V , namely $-\mathbf{V} = \text{Diag}(V - V \cdot \mathbf{I})$, as follows:

Let $H := F^{-T} \cdot I \cdot F^{-1}$ and $A := F^{-T} \cdot V \cdot F^{-1}$ though neither is likely to be computed. Since $F^T \cdot (H - \bar{H}) \cdot F = I$ and $F^T \cdot (A - \bar{A}) \cdot F = V$, the eigenvalues v in V and eigenvectors \mathbf{f} in F belong to perturbed matrices $H - \bar{H}$ and $A - \bar{A}$, so the same derivation as produced estimates

for $\mathbf{e}^T \cdot (\mathbf{A} - \mathbf{V} \cdot \mathbf{H}) \cdot \mathbf{e}$ on the previous page now produces a column of first-order estimates:
 $\text{diag}(\mathbf{V} - \mathbf{V} \cdot \mathbf{I}) \text{diag}(\mathbf{F}^T \cdot \mathbf{A} \cdot \mathbf{F} - \mathbf{V} \cdot \mathbf{F}^T \cdot \mathbf{H} \cdot \mathbf{F}) = \mathbf{v} := \text{diag}(\mathbf{V} - \mathbf{V} \cdot \mathbf{I})$ ignoring $(\dots)^2$ terms.

Alas, eigenvalues repeated or clustered too closely invalidate the foregoing estimate's derivation. Then complicated estimates like those in ch. VI §3 of the book by Stewart & Sun (1990) are afflicted by "difficult and unresolved problems". Their complications will be left unexplored here because, among other things, without extra-precise arithmetic the residuals \mathbf{I} and \mathbf{V} can rarely be computed accurately enough to provide reliable estimates \mathbf{v} of $\text{diag}(\mathbf{V} - \mathbf{V} \cdot \mathbf{I})$.

Though roundoff contaminates the residuals, some of them badly, they remain worth computing to assess the adequacy of a program's computed eigenvectors \mathbf{F} . In so far as these are usually computed only to provide a new coordinate system that uncouples the natural modes of an elastic structure's vibrations, usually almost all that matters is how nearly \mathbf{F} diagonalizes \mathbf{A} and \mathbf{H} . Tiny residuals $\mathbf{I} := \mathbf{F}^T \cdot \mathbf{H} \cdot \mathbf{F} - \mathbf{I}$ and $\mathbf{V} := \mathbf{F}^T \cdot \mathbf{A} \cdot \mathbf{F} - \mathbf{V}$, where $\mathbf{V} = \text{Diag}(\mathbf{F}^T \cdot \mathbf{A} \cdot \mathbf{F})$, are about as tiny as we can reasonably expect from the program if the computed residuals are not much bigger than the uncertainties they acquire from roundoff when computed with the same arithmetic as the program's. (We also expect no column of \mathbf{F} to be unnecessarily big; more about this shortly.)

How much uncertainty must these residuals inherit from the act of computing them? Let's see ...

The product $\mathbf{P} := \mathbf{B} \cdot \mathbf{C}$ of two n -by- n matrices is obscured by roundoff from floating-point scalar multiplications and additions. The multiplications' rounding errors are easiest to over-estimate:

Were only additions performed exactly (not rounded off), the computed \mathbf{P} would satisfy simply $|\mathbf{P} - \mathbf{B} \cdot \mathbf{C}| \leq |\mathbf{B}| \cdot |\mathbf{C}| \cdot \epsilon / 2$ elementwise, where ϵ is a roundoff threshold like MATLAB's `eps`.

Rounded additions contribute additional uncertainty dependent upon the order in which additions are performed. For large n the simplest bound that is not outrageously pessimistic (but not necessarily the smallest uncertainty) is obtained from a divide-and-conquer order of additions that form a binary tree; see (4.6) on p. 83 of N. Higham's (2002) book. The resulting estimate is

$$|\mathbf{P} - \mathbf{B} \cdot \mathbf{C}| \leq |\mathbf{B}| \cdot |\mathbf{C}| \cdot (1 + \lceil \log_2(n) \rceil) \cdot \epsilon / 2 \text{ elementwise}$$

if terms of order ϵ^2 are ignored. Judicious applications of this inequality define rough estimates of the intrinsic uncertainty each residual will be deemed to inherit from the act of computing it:

Computed Residual	Its Approximated Uncertainty
$\mathbf{I} := \mathbf{F}^T \cdot (\mathbf{H} \cdot \mathbf{F}) - \mathbf{I}$	$\overline{\mathbf{I}} := \mathbf{F}^T \cdot (\mathbf{H} \cdot \mathbf{F} + \mathbf{H} \cdot \mathbf{F}) \cdot (1 + \lceil \log_2(n) \rceil) \cdot \epsilon / 2$
$\mathbf{V} := \mathbf{F}^T \cdot (\mathbf{A} \cdot \mathbf{F}) - \mathbf{V}$	$\overline{\mathbf{V}} := \mathbf{F}^T \cdot (\mathbf{A} \cdot \mathbf{F} + \mathbf{A} \cdot \mathbf{F}) \cdot (1 + \lceil \log_2(n) \rceil) \cdot \epsilon / 2$
$\mathbf{R} := \mathbf{A} \cdot \mathbf{F} - \mathbf{H} \cdot (\mathbf{F} \cdot \mathbf{V})$	$\overline{\mathbf{R}} := (\mathbf{A} \cdot \mathbf{F} + \mathbf{H} \cdot \mathbf{F} \cdot \mathbf{V}) \cdot (2 + \lceil \log_2(n) \rceil) \cdot \epsilon / 2$
$\mathbf{v} := \text{diag}(\mathbf{V} - \mathbf{v} \cdot \text{diag}(\mathbf{I}))$	$\overline{\mathbf{v}} := \text{diag}(\overline{\mathbf{V}}) + \mathbf{v} \cdot \text{diag}(\overline{\mathbf{I}})$

Here diagonal matrix $\mathbf{V} = \text{Diag}(\mathbf{F}^T \cdot \mathbf{A} \cdot \mathbf{F})$ is stored typically as a column $\mathbf{v} := \text{diag}(\mathbf{V})$ intended to approximate the desired eigenvalues' $\text{diag}(\mathbf{V})$. Its first-order error-estimate \mathbf{v} inherits from the residuals its uncertainty $\overline{\mathbf{v}}$, both computed elementwise from the formulas tabulated above.

Given $\mathbf{A}, \mathbf{H}, \mathbf{F}$ and \mathbf{v} , a MATLAB program `rsdls.m` listed in §11 automatically determines an appropriate \mathbf{I} and computes $\mathbf{I}, \mathbf{V}, \overline{\mathbf{I}}, \overline{\mathbf{V}}, \mathbf{v}$ and $\overline{\mathbf{v}}$ from formulas like those tabulated.

Those uncertainties $\overline{\dots}$ tend to over-estimate the contributions of roundoff very pessimistically, as if the rounding errors had conspired to achieve their worst imaginable effect. Rounding errors usually seem random and uncorrelated (though actually they are not); and if rounding conforms to the default specifications of *IEEE Standards 754 & 854* then rounding errors seem unbiased too, which inclines the *Law of Averages* more towards cancellation than reinforcement of errors. Consequently those approximated uncertainties $\overline{\dots}$ tend usually to be too big by factors roughly as big as $\sqrt{12 \cdot n}$ for a product of two n -by- n matrices, and $\sqrt{24 \cdot n}$ for a product of three, unless some small minority of the intermediate terms in those products are much bigger than the rest.

Comparisons of computed residuals each with its intrinsic uncertainty, despite their pessimism, seem at least plausibly to indicate how adequately a program has performed its intended task:— computation of eigenvector estimates F that make all three residuals negligible. Therefore ...

Each computed residual will be divided by its overestimated uncertainty elementwise, and a large quotient shall be deemed a sure signal of inadequate accuracy in computed results.

“Inadequate Accuracy” will be signaled too frequently if our definition of intrinsic uncertainty is too small, and too infrequently if too big. Alas, this signal can get muffled, misleading us to accept results far worse than the data deserve, by a rare phenomenon that generates excessively big residuals and their alleged uncertainties:

The computed F can have some unnecessarily big column(s).

Any program intended to solve the general eigenproblem can be undermined by this phenomenon. It can happen only when A and H , though scaled to make $\text{Diag}(H) = I$, share a near nullspace. Then accurate eigenvectors may appear to be inaccessible because, if a vector $\mathbf{z} \neq \mathbf{0}$ in that near nullspace satisfies both $A \cdot \mathbf{z} = \mathbf{0}$ and $H \cdot \mathbf{z} = \mathbf{0}$, almost arbitrary multiples of \mathbf{z} can be added to all different approximate eigenvectors in F without much altering $F^T \cdot H \cdot F$ nor $F^T \cdot A \cdot F$. If this phenomenon is detected, some of the contamination by \mathbf{z} of a computed eigenvector \mathbf{f} can be removed from \mathbf{f} by replacing it by $\mathbf{g} := \mathbf{f} - \mathbf{z} \cdot (\mathbf{z}^T \cdot H \cdot \mathbf{f}) / (\mathbf{z}^T \cdot H \cdot \mathbf{z})$ provided doing so does not shrink $\mathbf{g}^T \cdot H \cdot \mathbf{g} = \mathbf{f}^T \cdot H \cdot \mathbf{f} - (\mathbf{z}^T \cdot H \cdot \mathbf{f})^2 / (\mathbf{z}^T \cdot H \cdot \mathbf{z})$ too much below $\mathbf{f}^T \cdot H \cdot \mathbf{f}$. Then the new matrix G of near-eigenvectors \mathbf{g} must be iteratively refined by a program like `gnsym eig` unlikely to resurrect the contamination.

How can this misleading phenomenon be detected? First it must be suspected. First suspicion is aroused by exceptionally big computed eigenvectors \mathbf{f} associated with intolerably big computed uncertainties among the elements of $\overline{\mathbf{v}}$ compared with the corresponding eigenvalues in \mathbf{v} . Among these big eigenvectors \mathbf{f} the ones belonging to comparatively small eigenvalues in \mathbf{v} seem most likely to serve as the previous paragraph's \mathbf{z} . Weasel-words like “exceptionally big”, “comparatively small” and “most likely” fail to define the phenomenon sharply. It hardly ever happens sharply, but it does happen.

For example, suppose the first three true eigenvalues in \mathbf{v} are repeated but otherwise separated well from the rest. Then the first three columns of E must be partially indeterminate in so far as they can be postmultiplied by any 3-by-3 orthogonal matrix $Q = Q^{-T}$ and still serve no less well than before as eigenvectors satisfying $\text{Diagonal} = E^T \cdot A \cdot E$, $E^T \cdot H \cdot E = I$, and $A \cdot E = H \cdot E \cdot \mathbf{v}$. In a world without perturbations Q would not matter. It can matter a lot in our perturbed world.

No matter what program computes them, if computed well the first three computed eigenvalues in V will be clustered closely and separated well from the others. The first three columns $[\mathbf{f}_1, \mathbf{f}_2, \mathbf{f}_3]$ of the computed F , though accidents of roundoff, will approximate three eigenvectors well. How nearly linearly dependent are they? Maybe too nearly if H is too nearly singular, in which case at least one of these columns will be bigger than necessary. The remedy is to replace these columns by $[\mathbf{f}_1, \mathbf{f}_2, \mathbf{f}_3] \cdot Q$ with an aptly chosen 3-by-3 orthogonal Q . Apt choices will yield three columns $[\mathbf{f}_1, \mathbf{f}_2, \mathbf{f}_3] \cdot Q$ of which about as few as necessary are extraordinarily big. The *Singular-Value Decomposition* $[\mathbf{f}_1, \mathbf{f}_2, \mathbf{f}_3] = P \cdot \Sigma \cdot Q^T$, wherein $P^T \cdot P = I$ and Σ are 3-by-3 diagonal matrices, offers an apt choice for Q . This replacement of unnecessarily big columns of F by smaller ones should precede both iterative refinement and the subsequent computations of residuals and alleged intrinsic uncertainties. Note that iterative refinement can order the cluster's eigenvalues and their eigenvectors differently than before, which can complicate comparisons of residuals and error estimates after refinement against before.

The remedy just proposed falls short of a prescription if close clusters of eigenvalues separated well enough from the rest are unobvious. Fortunately the remedy is needed extremely rarely. If the need is obvious the proposed remedy can reduce by orders of magnitude the norms of some computed eigenvectors and the the errors and alleged uncertainties in their computed eigenvalues.

The following 6-by-6 example was run under MATLAB 6.5 on IBM T21 and Dell OptiPlex GX400 Wintel computers to illustrate that ...

- The need for the foregoing remedy is an unlikely accident of roundoff that cannot occur unless H is very nearly singular and shares a near nullspace with A .
- If suspected, the need can be confirmed most quickly by applying the remedy.
- If needed, the remedy attenuates some errors enormously, and all the more so when matrix multiplications are accumulated extra-precisely.

The example's $A := G^T \cdot \text{Diag}(\mathbf{a}) \cdot G$ and $H := G^T \cdot \text{Diag}(\mathbf{h}) \cdot G$ are generated exactly as matrices of big integers from this given data:

Matrix G						Column a	Column h
2	3	9	5	-16	17	1436714424605	1041044474703
-9	-3	18	16	14	-12	0	1502331013996
-10	-11	14	-17	0	-13	1527439170635	293126770298
-18	0	-1	-8	14	9	0	1020518759025
-4	8	15	7	10	-7	0	1
10	14	-18	3	6	-1	-1694061335945	1278112860186

Each element of A and H is an accumulation of 14-digit integers; no element exceeds $1043661783423569 < 2^{50}$ in magnitude. Therefore no rounding errors occurred during their computation in MATLAB's arithmetic carrying 53 sig. bits, so all six true eigenvalues can be computed as $\hat{\mathbf{e}} := \text{sort}(\mathbf{a}/\mathbf{h} \text{ elementwise})$ accurate to fully 53 sig. bits:

$\hat{\mathbf{e}}^T :$	-1.3254395513229311	0	0	0	1.3800701694467767	5.2108484294428896
------------------------	---------------------	---	---	---	--------------------	--------------------

The three repeated eigenvalues chosen to be zeros were so chosen to make their errors easier to appreciate. At least one of their errors must be big because H so nearly annihilates at least one of their three eigenvectors. H has a huge condition number $\|H\| \cdot \|H^{-1}\| = 8.49e14$; after H is diagonally scaled to make its diagonal I its condition number drops slightly to $6.74e14$. This compels at least one eigenvector to be relatively huge; just one is. All eigenvectors

are hypersensitive to roundoff-like perturbations because A and H so nearly share a nullspace. Eigenvectors were computed in extra-wide arithmetic from the formula $\hat{E} := G^{-1} \cdot \text{Diag}(1/\bar{h})$ elementwise, and then the columns of \hat{E} were reordered to match the sorted eigenvalues, rounded off, and stored. These eigenvector columns are accurate to the last (53rd) sig. bit or two. Their squared norms turned out to be respectively ...

$\text{sum}(\hat{E} \cdot \hat{E}^T) :$	3.1418e-13	1.4043e-13	6.6622e-15	0.2246	1.8611e-13	6.4182e-13
---	------------	------------	------------	---------------	------------	------------

... which indicate that the fourth eigenvalue $\hat{e}(4)$ may be hypersensitive to roundoff-like perturbations. To check this out, MATLAB program `rsdls.m` employing the formulas for “Computed Residual” and “Its Approximated Uncertainty” tabulated above (this program is listed in §11) computed residuals I and \hat{E} and first-order error-estimate \hat{e} along with respective uncertainties \bar{I} , $\bar{\hat{E}}$ and $\bar{\hat{e}}$ inherited from matrix multiplications’ roundoff, but using \hat{E} and $\hat{E} := \text{Diag}(\hat{e})$ in place of the tabulated formulas’ F and $V = \text{Diag}(v)$. By default MATLAB 6.5’s arithmetic is rounded to 53 sig. bits; this arithmetic produced $|I| < 0.0791 \cdot \bar{I}$ and $|\hat{E}| < 0.135 \cdot \bar{\hat{E}}$ elementwise. Residuals that much tinier than their uncertainties were likely to be drowned in their own rounding errors. The same went for the utterly wrong first-order correction \hat{e} and its uncertainty $\bar{\hat{e}}$ inherited from roundoff:

$\hat{e}^T :$	3.9968e-15	-1.4178e-15	-5.2676e-17	-3.4787e-3	1.1102e-15	4.0856e-14
$\bar{\hat{e}}^T :$	4.6432e-13	6.5457e-14	1.0436e-15	9.6749e-2	2.7184e-13	2.702e-12

MATLAB 6.5’s `system_dependent('setprecision'64)` accumulated every subsequent matrix multiplication to 64 sig. bits before rounding it back to 53 to get residuals and a first-order correction more nearly deserved by data \hat{E} and \hat{e} . Their elementwise $\max |I|/\bar{I} = 29.05$ and $\max |\hat{E}|/\bar{\hat{E}} = 11.0$. Consequently the first several binary digits of some residuals became worth having; and the same went for some much-improved first-order corrections:

$\hat{e}^T :$	6.6613e-16	-3.1656e-19	6.4748e-21	3.7334e-7	0	-1.7764e-15
$\bar{\hat{e}}^T :$	4.2633e-15	3.1962e-17	5.0958e-19	4.7629e-5	3.0039e-15	1.2736e-14

Now we shall not be surprised by huge errors and/or uncertainties afflicting at least one of the three computed values of tiny eigenvalues $\hat{e}(2:4)$ because now we know how big are an eigenvector and its contribution to uncertainty.

Rarely would the foregoing facts about the eigensystem and H ’s condition numbers be known in advance. They have been exposed here to help readers decide what to scrutinize among the following voluminous computations.

MATLAB 6.5’s default mode, `system_dependent('setprecision'53)` that rounds all arithmetic to 53 sig. bits, was in force for all results below until declared otherwise. MATLAB commands `[F, V] = eig(A, H)` and `v = diag(V)` computed approximate eigenvectors F and a column of eigenvalues v whose errors are displayed here above the squared norms of respective eigenvectors:

$(\hat{e} - v)^T :$	3.1086e-15	6.9111e-10	2.1650e-15	-6.7233e-10	-2.2204e-16	2.0428e-14
$\text{sum}(F \cdot F^T) :$	3.1629e-13	0.11447	1.9608e-14	0.11135	1.8699e-13	6.4444e-13

Two of the computed eigenvalues appear to be about eighteen sig. bits less accurate than the others; they correspond to two computed eigenvectors enormously bigger than the others, which are all a little too big. How can anyone assess whether both of the two enormous vectors deserve to be so much bigger than the others? It’s not obvious.

Residuals I and V seem appropriately small compared with their uncertainties \bar{I} and \bar{V} computed from the aforementioned tabulated formulas: $|I| < 0.086 \cdot \bar{I}$ and $|V| < 0.31 \cdot \bar{V}$ elementwise. These inequalities arouse no suspicions that parts of `eig`’s results F and V are far worse than they have to be. On the contrary, the residuals are sufficiently smaller than their uncertainties that we might reasonably expect many of `eig`’s eigenvalues’ errors $\hat{e} - v$ and their first-order error estimates \bar{v} (almost all blighted by roundoff during their computation) to be at least an order of magnitude tinier than their crude uncertainties \bar{v} . Most are much tinier:

$(\hat{e} - v)^T :$	3.1086e-15	6.9111e-10	2.165e-15	-6.7233e-10	-2.2204e-16	2.0428e-14
$v^T :$	1.0214e-14	1.5868e-3	2.3777e-15	-1.716e-3	-1.3545e-14	8.1712e-14
$\bar{v}^T :$	4.6744e-13	4.9713e-2	7.7406e-15	4.8362e-2	2.7316e-13	2.7134e-12

Rarely would we know that. Instead we might reasonably infer that two eigenvalues are extremely hypersensitive to roundoff-like perturbations. Can iterative refinement improve their accuracy? New eigenvalue estimates \mathbf{w} obtained from $[Z, \mathbf{w}] = \text{gnsymeig}(F' * A * F, F' * H * F)$ have errors $\hat{\mathbf{e}} - \mathbf{w}$ compared here with the old errors $\hat{\mathbf{e}} - \mathbf{v}$:

new $(\hat{\mathbf{e}} - \mathbf{w})^T$:	-7.1054e-15	1.718e-3	-7.9518e-17	-1.5893e-3	1.3323e-14	-6.1284e-14
old $(\hat{\mathbf{e}} - \mathbf{v})^T$:	3.1086e-15	6.9111e-10	2.1650e-15	-6.7233e-10	-2.2204e-16	2.0428e-14

Most of the “refined” estimates \mathbf{w} are *worse* than the old because 53 sig. bit roundoff has contaminated $F^T \cdot H \cdot F$ and $F^T \cdot A \cdot F$ badly, and more so since one column of F is unnecessarily big. The remedy described above is needed:

Three computed eigenvalues $\mathbf{v}(2:4)$ constitute a tight cluster separated well from the others. MATLAB’s compact singular value decomposition $[P, \text{Phi}, Q] = \text{svd}(F(:, 2:4), 0)$ supplied a 3-by-3 orthogonal matrix Q which was used to replace the cluster’s three computed eigenvectors $F(:, 2:4)$ by $F(:, 2:4) \cdot Q$ to obtain the remedied matrix $\text{Fr} = [F(:, 1), F(:, 2:4) * Q, F(:, 5:6)]$ of eigenvectors. Comparing the old squared norms with the new ...

old $\text{sum}(F * F)$:	3.1629e-13	0.11447	1.9608e-14	0.11135	1.8699e-13	6.4444e-13
new $\text{sum}(\text{Fr} * \text{Fr})$:	3.1629e-13	0.22582	1.8248e-15	1.2781e-15	1.8699e-13	6.4444e-13

... suggests that only one of the three eigenvectors has to be big. Cancellation has left few of the other two’s leading digits uncorrupted by roundoff. Consequently some residuals in $\bar{\mathbf{I}}$ and $\bar{\mathbf{V}}$ recomputed using Fr in place of F are now huge compared with their recomputed uncertainties $\bar{\mathbf{I}}$ and $\bar{\mathbf{V}}$; the biggest of the ratios $|\bar{\mathbf{I}}| / \bar{\mathbf{I}}$ elementwise has grown to $8.02e5$, and of $|\bar{\mathbf{V}}| / \bar{\mathbf{V}}$ to $9.93e5$. Thus has remediation reduced two eigenvectors’ magnitudes and also their amplification of roundoff in the residuals, some of which now rise high enough above their noise to justify recomputation of the first-order correction \mathbf{v} , especially $\mathbf{v}(3:4)$:

$(\hat{\mathbf{e}} - \mathbf{v})^T$:	3.1086e-15	6.9111e-10	2.1650e-15	-6.7233e-10	-2.2204e-16	2.0428e-14
\mathbf{v}^T :	1.0214e-14	1.4499e-3	2.1339e-15	-6.7233e-10	-1.3545e-14	8.1712e-14
$\bar{\mathbf{v}}^T$:	4.6744e-13	9.6958e-2	7.8532e-16	6.7731e-16	2.7316e-13	2.7134e-12

Now $\bar{\mathbf{v}}(3:4)$ are so much smaller than $\mathbf{v}(3:4)$ that $\mathbf{v}(3:4) + \bar{\mathbf{v}}(3:4)$ are almost surely more accurate than $\mathbf{v}(3:4)$. To improve the accuracies also of the remedied eigenvectors in Fr we must invoke iterative refinement:

$[Z, \mathbf{w}] = \text{gnsymeig}(\text{Fr}' * A * \text{Fr}, \text{Fr}' * H * \text{Fr})$ has errors $\hat{\mathbf{e}} - \mathbf{w}$ compared here with the old errors $\hat{\mathbf{e}} - \mathbf{v}$:

old $(\hat{\mathbf{e}} - \mathbf{v})^T$:	3.1086e-15	6.9111e-10	2.1650e-15	-6.7233e-10	-2.2204e-16	2.0428e-14
new $(\hat{\mathbf{e}} - \mathbf{w})^T$:	-6.6613e-15	6.4616e-17	2.0476e-17	-1.4499e-3	1.2879e-14	-6.1284e-14

Again “refinement” has slightly worsened most of the eigenvalues; but besides reordering the three tiny eigenvalues it has improved two of them by orders of magnitude. The third tiny eigenvalue seems not so tiny any more; it has been inflated by rounding errors in $\text{Fr}^T \cdot H \cdot \text{Fr}$ and $\text{Fr}^T \cdot A \cdot \text{Fr}$ amplified by Fr ’s one big column. Refined remedied eigenvector matrix $\text{Frz} := \text{Fr} \cdot Z$ was substituted for F , and $\mathbf{W} := \text{Diag}(\mathbf{w})$ for \mathbf{V} , in the tabulated formulas above to recompute Frz ’s squared norms, residuals $\bar{\mathbf{I}}$ and $\bar{\mathbf{W}}$, first-order eigenvalue error-estimates $\bar{\mathbf{w}}$, and their uncertainties $\bar{\bar{\mathbf{I}}}$, $\bar{\bar{\mathbf{W}}}$ and $\bar{\bar{\mathbf{w}}}$ resp. to get $|\bar{\mathbf{I}}| \approx 0.21 \cdot \bar{\mathbf{I}}$, $|\bar{\mathbf{W}}| \approx 0.62 \cdot \bar{\mathbf{W}}$ and ...

$\text{sum}(\text{Frz} * \text{Frz})$:	3.1073e-13	1.5264e-15	4.2094e-15	0.22325	1.8333e-13	6.4133e-13
actual $(\hat{\mathbf{e}} - \mathbf{w})^T$:	-6.6613e-15	6.4616e-17	2.0476e-17	-1.4499e-3	1.2879e-14	-6.1284e-14
computed $\bar{\mathbf{w}}^T$:	-2.3315e-14	6.0208e-18	-6.7974e-18	3.1025e-3	1.5099e-14	-1.8741e-13
its uncertainty $\bar{\bar{\mathbf{w}}}^T$:	4.5921e-13	8.8346e-16	1.4754e-15	9.7605e-2	2.6769e-13	2.6998e-12

Again, the residuals are sufficiently smaller than their uncertainties to be mostly overwhelmed by roundoff, so their derived estimate $\bar{\mathbf{w}}$ of the eigenvalues’ error $\hat{\mathbf{e}} - \mathbf{w}$ is far tinier than its crude uncertainty $\bar{\bar{\mathbf{w}}}$ and hence almost all wrong, submerged in roundoff. $\hat{\mathbf{e}} - \mathbf{w}$ is wrong mainly because $\text{gnsymeig}(\text{Fr}' * A * \text{Fr}, \text{Fr}' * H * \text{Fr})$ got the wrong matrices to refine; roundoff accumulating during the matrix multiplications $\text{Fr}^T \cdot A \cdot \text{Fr}$ and $\text{Fr}^T \cdot H \cdot \text{Fr}$ vitiated the processes of remediation and refinement. 53 sig. bits are too few to compute all residuals reliably enough.

No remediation but iterative refinement was performed to improve the accuracies of the eigenvector matrix F_x ; the results from $[Z_x, w_x] = \text{gnsymeig}(F_x' * A * F_x, F_x' * H * F_x)$, and `rsdls` applied to $F_x Z_x$ and $W_x := \text{Diag}(w_x)$ instead of F and V , benefited from matrix multiplication's 11 extra sig. bits. Now elementwise $\max | | / \bar{I} = 53.7$ and $\max | V_x | / \bar{W}_x = 21.2$ and the other results ...

error $(\hat{e} - w_x)^T$:	2.2204e-16	1.4385e-19	-2.8943e-19	-6.6453e-7	8.8818e-16	-1.7764e-15
estimate w_x^T :	4.4409e-16	9.8384e-20	-3.139e-19	1.742e-6	1.3323e-15	-2.6645e-15
its uncertainty $\overline{w_x^T}$:	4.2635e-15	1.0788e-18	3.2414e-19	4.7629e-5	3.0027e-15	1.2737e-14

... have roughly the same accuracies as the results u , \bar{u} and \overline{u} obtained after remediation.

Thus the foregoing example supports the conclusion that, after the unlikely need for a remedy has been detected, our remedy for unnecessarily big computed eigenvectors undoes their ill effects. Another conclusion is that 11 extra sig. bits carried during matrix multiplication sufficiently suppress the noises that beset numerical signals to ease greatly the task of delivering at least about as much accuracy as the data deserve, but we all already knew that; see my web page's posting at [<.../MxMuleEps.pdf>](#).

What we do not know is how often, in practice, results from programs like $\text{eig}(A, H)$ need remediation nor, without it, how often and how badly those results mislead their consumers. Nobody is keeping score; to do so would require at least the computation of residuals and their uncertainties by a program like `rsdls`, and possibly refinement by a program like `gnsymeig`. We could regard these supererogatory computations' costs as premiums paid for insurance against dire consequences of being misled by misleading results, but only if the risks were quantified. Usually they aren't. Instead, in the current climate, programs like `rsdls` and `gnsymeig` will be employed occasionally to evince the exercise of *Due Diligence*, or to diagnose suspicious results. Suspicions might be aroused by a computed eigenvector matrix more ill-conditioned (closer to singular) than expected, or by a symmetry (repeated eigenvalues) unexpectedly broken, or by an unexpected bump in computed results' variation as a parameter changes, or by a *buzz* (high-frequency vibration) of a structural component whose secure attachment had been overlooked, if anyone is looking now.

§9: How Well Does `gnsymeig` Work?

How do `gnsymeig`'s residuals compare with their inherited uncertainties? This question is being explored for a wide range of input data $\{A, H\}$.

TO BE CONTINUED.

§10: References:

D. Corneil (1965) *Eigenvalues and Orthogonal Eigenvectors of Real Symmetric Matrices*, MSc. Thesis, Computer Science Dept., University of Toronto, Canada.

G.H. Golub & C.F. Van Loan (1996) *Matrix Computations* 3d. ed.: Johns Hopkins Univ. Press.

N.J. Higham (2002) *Accuracy & Stability of Numerical Algorithms* 2nd ed.: Soc. for Indust. & Appl. Math., Philadelphia (esp. ch. 4 for *Pairwise Summation* and *Compensated Summation*)

W. Kahan (2004) *MATLAB's Loss is Nobody's Gain*
<www.cs.berkeley.edu/~wkahan/MxMulEps.pdf> esp. pp. 16 - 27.

————— (2006) *How Futile are Mindless Assessments of Roundoff in Floating-Point Computation ?* <www.cs.berkeley.edu/~wkahan/Mindless.pdf> esp. §§14 - 15.

————— (2008) *Computing Cross-Products and Rotations in 2- and 3-Dimensional Euclidean Spaces* <www.cs.berkeley.edu/~wkahan/MathH110/Cross.pdf>

————— (2008) *Do MATLAB's lu(...), inv(...), / and \ have Failure Modes ?*
<www.cs.berkeley.edu/~wkahan/Math128/FailMode.pdf>

B.N. Parlett (1998) *The Symmetric Eigenvalue Problem* (Classics in Applied Mathematics #20): Soc. for Indust. & Appl. Math., Philadelphia (esp. ch. 15 for the general symmetric eigenproblem)

I. Slapničar & V. Hari (1991) “On the Quadratic Convergence of the Falk-Langemeyer Method” pp. 84-114 of *SIAM Jl. of Matrix Anal. & Appl.* **12**. A method very similar to ours treated by them is traced back to publications in 1960 and 1965; none proved global convergence.

A. van der Sluis (1969) “Condition Numbers and Equilibration of Matrices” pp. 14 - 23 of *Numerische Mathematik* **14**. Proved also in §7.3 of Higham's (2002) book.

G.W. Stewart & J-g. Sun (1990) *Matrix Perturbation Theory*: Academic Press, San Diego. Many of its errors are corrected at <ftp://ftp.cs.umd.edu/pub/stewart/errata/pert.ps>.

J.H. Wilkinson (1965) *The Algebraic Eigenvalue Problem*, Oxford University Press (pp. 265-282)


```

dE = sum(abs( E(:) - [1; 0; 0; 1] )) ; % = || E - I ||
bigE = (dE > eps2) ; % Is ||E - I|| big enough to matter?
if bigE, % Yes; ||E - I|| is big enough!
%   Select columns that E will affect:
    hij = H(:,[i,j]) ; aij = A(:,[i,j]) ;
    end % of computing congruence E when H isn't nearly singular

else % Cope with rare near-singular H ( 0.75 <= sa < 1 ) :
    costheta2 = (1 - sa)*(1 + sa) ; s = sign(sigma) ;
%   Select columns that E will affect:
    hij = H(:,[i,j]) ; aij = A(:,[i,j]) ;
%   Apply congruence X that will cancel critical data exactly:
    X = [1, s; s, -1] ; F(:,[i,j]) = F(:,[i,j])*X ;
    hij = hij*X ; hij([i,j],:) = X*hij([i,j],:) ;
    aij = aij*X ; aij([i,j],:) = X*aij([i,j],:) ;
%   Scale Diag(H) back to I :
    dii = hij(i,1) ; djj = hij(j,2) ;
    aii = aij(i,1)/dii ; ajj = aij(j,2)/djj ;
    dii = sqrt(dii) ; djj = sqrt(djj) ;
    hij(:,1) = hij(:,1)/dii ; hij(:,2) = hij(:,2)/djj ;
    hij([i,j],:) = eye(2) ; t = aij(i,2)/(dii*djj) ;
    aij(:,1) = aij(:,1)/dii ; aij(:,2) = aij(:,2)/djj ;
    aij([i,j],:) = [aii,t; t,ajj] ; psi = (aii - ajj)*0.5 ;
    F(:,i) = F(:,i)/dii ; F(:,j) = F(:,j)/djj ;
%   Compute angle for diagonalizing congruence by reflection E :
    if (t == 0), psi = 0 ;
        elseif (psi == 0), psi = 0.5*piby2 ;
            else psi = atan(t/psi)*0.5 ; end
    t = sign(t) ; t = (t*s - 1)*t*0.5 ;
    if (t == 0), t = 1 ; else t = t*im ; end
    t = t*exp(im*psi) ; co = real(t) ; si = imag(t) ;
    E = [co, si; si, -co] ; kbigH = kbigH + 1 ;
    end % of computing congruence E when H is nearly singular.

if bigE % Perform a substantial 2-by-2 congruence E'*(__)*E :
    kstp = kstp + 1 ;
    hij = hij*E ; hij([i,j],:) = eye(2) ;
    aij = aij*E ; aij([i,j],:) = E'*aij([i,j],:) ;
    aij(i,2) = 0 ; aij(j,1) = 0 ; %... clear out rounding errors
%   Mitigate rounding errors in tinier eigenvalue:
    if (abs(aij(i,1)) < 0.125*abs(aij(j,2)))
        t = [v1, -alpha]*[v2; alpha] ;
        aij(i,1) = t/(aij(j,2)*costheta2) ;
    elseif (abs(aij(j,2)) < 0.125*abs(aij(i,1)))
        t = [v1, -alpha]*[v2; alpha] ;
        aij(j,2) = t/(aij(i,1)*costheta2) ;
    end % of adjusting tinier eigenvalue
%   Undo occasional disorder between new diagonal entries of A :
    if (aij(i,1) > aij(j,2)) % happens only if old entries were ==
        aij = fliplr(aij) ; hij = fliplr(hij) ; E = fliplr(E) ;
        aij([i,j],:) = aij([j,i],:) ; hij([i,j],:) = hij([j,i],:) ;
    end
%   Update rows and columns of A and H , and columns of F :
    Aij = A(:,[i,j]) ; A(:,[i,j]) = aij ; A([i,j],:) = aij' ;

```



```

aE = abs(E) ; aHE = abs(HE) ; aAE = abs(AE) ;
eHe1 = E'*HE ; %... but E might not have been scaled yet ...
dh = diag(eHe1) ; sdh = sqrt(dh) ; Sdh = sdh*sdh' ;
Sdh = (Sdh - diag(diag(Sdh))) + diag(dh) ; %... scale factors
% Now scaled(E'*X*E) = E'*X*E./Sdh for any X ...

eHe = eHe1./Sdh - eye(n) ; eAee = E'*AE ;
eAe = eAee./Sdh - diag(e) ;
de = diag(eAe) ; % ... = diag(eAe) - e.*diag(eHe)

% What are the roundoff thresholds? First for scalars:
eeps = ((n+4)-n)/3 ; %... to defeat compiler "optimization"
eeps = abs((eeps - 1)*3 - 1) ; %... == eps ?
eeps = max(eeps, eps) ; %... MATLAB 7+'s eeps can be weird.
% Next for matrix multiplication:
eta = 0.5^32 ; lreta = [-1, eta, 1]*[1; eta; 1] ; %... == 0 ?
% Most versions of MATLAB accumulate left-to-right, but ...
eta = [1, eta, -1]*[1; eta; 1] ; %... == lreta ?
if (eta ~= lreta), eta = max(lreta, eta) ; end %... eta ~= 0 .

if (eta == 0)|(eeps > eps) %... eeps is THE roundoff threshold
    c = ( 1 + log2n )*eeps*0.5 ;
    cEHE = aE'*(abs(H)*aE + aHE)*c ;
    cEAE = aE'*(abs(A)*aE + aAE)*c ;
else %... eps for scalars, eta for matrix multiplication:
    c = ( 1 + log2n )*eta ;
    cEHE = aE'*(abs(H)*aE+aHE)*c + (aE'*aHE+abs(eHe1))*eps*0.5 ;
    cEAE = aE'*(abs(A)*aE+aAE)*c + (aE'*aAE+abs(eAee))*eps*0.5 ;
end
cEHE = cEHE./Sdh ; cEAE = cEAE./Sdh ;
De = diag(cEAE) + abs(e).*diag(cEHE) ;

% = = = = =

```