Introduction

Can Householder tridiagonalization be used as preprocessing before the Jacobi algorithm?

For dense symmetric eigenvalue problems, standard solvers usually reduce the matrix to tridiagonal
and then use a tridiagonal eigensolver, such as implicit QR iteration or divide-and-conquer.

Jacobi nevertheless remains important because its rotations offer substantial parallelism,
and GPU libraries such as NVIDIA cuSOLVER provide dedicated Jacobi eigensolvers.

The usual objection to Householder preprocessing is that tridiagonalization costs $O(n^3)$,
and the Jacobi rotations that follow do not preserve the tridiagonal zeros.
It is therefore assumed that the preprocessing cannot improve the subsequent Jacobi computation.

While the factual statements are true, the assumption does not follow from them.

Destroying the zeros does not imply that the transformed starting matrix
has no effect on subsequent Jacobi convergence.

The effect also need not be the same for every matrix.
We therefore study several matrix families and examine algebraically
which properties make Householder preprocessing favorable or unfavorable.

The resulting predictions are then tested experimentally using measured sweep counts and runtimes.

Off-diagonal and diagonal energy (E) after tridiagonalization

Ⓐ

The first step of symmetric Householder tridiagonalization proceeds as follows:

$$\text{Partition } A = A^T \text{ as } \left[\begin{array}{c|c} a_{1,1} & \vec{x}^T \\ \hline \vec{x} & C \end{array}\right] \qquad \vec{x} = \begin{bmatrix} a_{2,1} \\ a_{3,1} \\ \vdots \\ a_{n,1} \end{bmatrix}$$

$$\text{Denote the squared length of the first column-tail } \vec{x} \text{ as } \rho^2 = \vec{x}^T \vec{x} = \sum_{i=2}^{n} \left(a_{i,1}{}^2\right)$$

$$\text{Choose the Householder matrix } Q_1 \text{ so that } Q_1 \vec{x} = \begin{bmatrix} \rho \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \rho \vec{e}_1, \qquad H_1 = \left[\begin{array}{c|c} 1 & \vec{0}^T \\ \hline \vec{0} & Q_1 \end{array}\right]$$

$$\text{① } B_1 = H_1 A = \left[\begin{array}{c|c} 1 & \vec{0}^T \\ \hline \vec{0} & Q_1 \end{array}\right] \left[\begin{array}{c|c} a_{1,1} & \vec{x}^T \\ \hline \vec{x} & C \end{array}\right] = \left[\begin{array}{c|c} a_{1,1} & \vec{x}^T \\ \hline \rho \vec{e}_1 & Q_1 C \end{array}\right]$$

$$② \; A_1 = B_1 H_1^{\intercal} = \left[\begin{array}{c|c} a_{1,1} & \vec{x}^{\intercal} \\ \hline \rho \vec{e}_1 & Q_1 C \end{array}\right]\left[\begin{array}{c|c} 1 & \vec{0}^{\intercal} \\ \hline \vec{0} & Q_1^{\intercal} \end{array}\right] = \left[\begin{array}{c|c} a_{1,1} & \rho \vec{e}_1^{\intercal} \\ \hline \rho \vec{e}_1 & Q_1 C Q_1^{\intercal} \end{array}\right]$$

- H_1 is orthogonal

↓

$$E\left(A_1\right) = E(A)$$

- E of the first column-tail of A becomes concentrated in $\left(A_1\right)_{2\,1}$
- E of the first row-tail of A becomes concentrated in $\left(A_1\right)_{1\,2}$
- Q_1 is orthogonal

↓

E of the trailing block C is equal to E of the trailing block $Q_1 C Q_1^{\intercal}$

Ⓑ

Now repeat the same construction one level down.

$$A_1 = \left[\begin{array}{c|c|c} a_{1,1} & \rho_1 & \vec{0}^{\intercal} \\ \hline \rho_1 & a_{2,2}' & \vec{y}^{\intercal} \\ \hline \vec{0} & \vec{y} & C_1 \end{array}\right] \qquad \vec{y} = \begin{bmatrix} a_{3,2}' \\ a_{4,2}' \\ \vdots \\ a_{n,2}' \end{bmatrix}$$

The second column-tail has squared length $\rho_2^{\,2} = \vec{y}^{\intercal}\,\vec{y} = \sum_{i=3}^{n} \left(\left(a'_{i,2}\right)^2\right)$

Choose Q_2 and H_2 so that $Q_2 \vec{y} = \begin{bmatrix} \rho_2 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$ $\quad H_2 = \begin{bmatrix} 1 & 0 & \vec{0}^T \\ 0 & 1 & \vec{0}^T \\ \vec{0} & \vec{0} & Q_2 \end{bmatrix}$

① $B_2 = H_2 A_1 = \begin{bmatrix} 1 & 0 & \vec{0}^T \\ 0 & 1 & \vec{0}^T \\ \vec{0} & \vec{0} & Q_2 \end{bmatrix} \begin{bmatrix} a_{1,1} & \rho_1 & \vec{0}^T \\ \rho_1 & a_{2,2}' & \vec{y}^T \\ \vec{0} & \vec{y} & C_1 \end{bmatrix} = \begin{bmatrix} a_{1,1} & \rho_1 & \vec{0}^T \\ \rho_1 & a_{2,2}' & \vec{y}^T \\ \vec{0} & \rho_2 \vec{e}_1 & Q_2 C_1 \end{bmatrix}$

② $A_2 = B_2 H_2^T = \begin{bmatrix} a_{1,1} & \rho_1 & \vec{0}^T \\ \rho_1 & a_{2,2}' & \vec{y}^T \\ \vec{0} & \rho_2 \vec{e}_1 & Q_2 C_1 \end{bmatrix} \begin{bmatrix} 1 & 0 & \vec{0} \\ 0 & 1 & \vec{0} \\ \vec{0}^T & \vec{0}^T & Q_2 \end{bmatrix} = \begin{bmatrix} a_{1,1} & \rho_1 & \vec{0}^T \\ \rho_1 & a_{2,2}' & \rho_2 \vec{e}_1^T \\ \vec{0} & \rho_2 \vec{e}_1 & Q_2 C_1 Q_2^T \end{bmatrix}$

As in step Ⓐ,

• H_2 is orthogonal

↓

$E(A_2) = E(A_1)$

• E of the second column-tail of A_1 becomes concentrated in $(A_2)_{3\,2}$

• E of the second row-tail of A_1 becomes concentrated in $(A_2)_{2\,3}$

• Q_2 is orthogonal

↓

E of the trailing block C_1 is equal to E of the trailing block $Q_2 C_1 Q_2^T$

©

Consider $A = A^{\mathsf{T}} = \left[\begin{array}{c|c|c} a_{11} & x & y \\ \hline x & a & b \\ \hline y & b & d \end{array}\right]$ with the trailing block $C = \left[\begin{array}{c|c} a & b \\ \hline b & d \end{array}\right]$

The first column-tail is

$$\vec{x} = (x, y)^{\mathsf{T}}, \qquad \rho = \sqrt{x^2 + y^2}$$

Choose Q so that $Q \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \rho \\ 0 \end{bmatrix} \rightarrow Q = \left[\begin{array}{c|c} \frac{x}{\rho} & \frac{y}{\rho} \\ \hline \frac{y}{\rho} & \frac{-x}{\rho} \end{array}\right]$

$$Q\,C\,Q^{\mathsf{T}} = \left[\begin{array}{c|c} \frac{a\,x^2 + 2\,b\,x\,y + d\,y^2}{\rho^2} & \frac{(a - d)\,x\,y + (y^2 - x^2)\,b}{\rho^2} \\ \hline \frac{(a - d)\,x\,y + (y^2 - x^2)\,b}{\rho^2} & \frac{a\,y^2 - 2\,b\,x\,y + d\,x^2}{\rho^2} \end{array}\right]$$

Its new off-diagonal entry is

$$b' = \frac{(a - d)\,x\,y + (y^2 - x^2)\,b}{\rho^2}$$

$$\text{Diagonal } E(A)' - \text{Diagonal } E(A) = 2\left(b^2 - (b')^2\right).$$

Therefore the diagonal E is unchanged exactly when

$$|b'| = |b|$$

$$\downarrow$$

$$|xy(a-d) + (y^2 - x^2)b| = (x^2 + y^2)|b|$$

$$\downarrow$$

$$|xy(a-d) + (y^2 - x^2)b| = |(x^2 + y^2)b|$$

$$\downarrow$$

- case 1:

$$xy(a-d) + (y^2 - x^2)b = (x^2 + y^2)b$$

- case 2:

$$xy(a-d) + (y^2 - x^2)b = -(x^2 + y^2)b$$

For $x \neq 0$ and $y \neq 0$, the two boundary conditions are

- $y(a-d) = 2xb$
- $x(a-d) = -2yb$

Diagonal E increases when the difference $(a-d)$ lies between the two boundary values

$$\min\left(\frac{2xb}{y}, \frac{-2yb}{x}\right) < (a-d) < \max\left(\frac{2xb}{y}, \frac{-2yb}{x}\right)$$

From the double inequality we can infer

- Favorable condition: $a \approx d$
- Unfavorable condition: $|a-d|$ is large relative to $|b|$

We therefore designed the test families to include both conditions.

Ⓓ

Quantitative summary

At every step i

- E of the column-i tail becomes concentrated in $(A_i)_{i+1,i}$
- E of the row-i tail becomes concentrated in $(A_i)_{i,i+1}$
- The trailing block C changes in ways that can move E either to or from the diagonal

How do we get the eigenvectors?

Denote the accumulated Householder transformation as H.
After tridiagonalization:

$$A' = H\,A\,H^T$$

Denote the accumulated Jacobi transformation as G.
After the Jacobi rotations, we get a diagonal matrix

$$D = G\,A'\,G^T = (G\,H)\,A\,(H^T\,G^T) = (G\,H)\,A\,(G\,H)^T$$

Define

$$Q = G\,H$$

$$\downarrow$$

$$D = Q\,A\,Q^T$$

↓

$$A\,Q^T = Q^T\,D$$

Since D is diagonal, each column of Q^T is multiplied by the corresponding diagonal entry of D.

Therefore

- The diagonal entries of D are the eigenvalues of A
- The columns of Q^T are the corresponding eigenvectors of A
- Thus the eigenvector matrix is

$$V = Q^T = H^T\,G^T$$

What can be processed simultaneously?

A Jacobi batch can contain only mutually nonintersecting pivot pairs.
Consider two Jacobi rotations R_1 and R_2:

- R_1 acts on the pivot pair (1, 3)
- R_2 acts on the pivot pair (2, 4)

These pairs are disjoint: they share no index.

$$R_1 = \begin{bmatrix} c_1 & 0 & s_1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ -s_1 & 0 & c_1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \qquad R_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & c_2 & 0 & s_2 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & -s_2 & 0 & c_2 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Because the nontrivial 2 × 2 blocks sit in disjoint rows and columns,

$$R_1 R_2 = R_2 R_1$$

$$R_1 R_2 = R_2 R_1 = \begin{bmatrix} c_1 & 0 & s_1 & 0 & 0 \\ 0 & c_2 & 0 & s_2 & 0 \\ -s_1 & 0 & c_1 & 0 & 0 \\ 0 & -s_2 & 0 & c_2 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Now consider an intersecting pair of rotations.

- S_1 acts on (1, 3)
- S_2 acts on (3, 4)

$$S_1 = \begin{bmatrix} c_1 & 0 & s_1 & 0 \\ 0 & 1 & 0 & 0 \\ -s_1 & 0 & c_1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad S_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & c_2 & s_2 \\ 0 & 0 & -s_2 & c_2 \end{bmatrix}$$

$$S_1 S_2 \neq S_2 S_1$$

$$S_1 S_2 = \begin{bmatrix} c_1 & 0 & s_1 c_2 & s_1 s_2 \\ 0 & 1 & 0 & 0 \\ -s_1 & 0 & c_1 c_2 & c_1 s_2 \\ 0 & 0 & -s_2 & c_2 \end{bmatrix} \qquad S_2 S_1 = \begin{bmatrix} c_1 & 0 & s_1 & 0 \\ 0 & 1 & 0 & 0 \\ -c_2 s_1 & 0 & c_1 c_2 & s_2 \\ s_1 s_2 & 0 & -c_1 s_2 & c_2 \end{bmatrix}$$

Ⓐ

Rotations acting on non-intersecting pivot pairs commute
and can be assembled into one batch matrix G
The batch update has the form $A_{new} = G A G^T$

Ⓑ

Rotations acting on intersecting pivot pairs do not commute
and cannot be processed simultaneously as one Jacobi batch

Ⓒ

Each batch can be computed in three stages:

① All rotations are computed from the same pre-batch matrix A in parallel.
② Left multiplication by all nonintersecting rotations is performed in parallel.
The updates are synchronized.
③ Right multiplication by all transposed rotations is performed in parallel.
The updates are synchronized before the next batch.

Color coding for the products:
- cyan: entries depending only on rotation 1
- red: entries depending only on rotation 2
- green: entries depending on both rotations

First Jacobi batch after tridiagonalization

Jacobi pivot selection has a tradeoff.

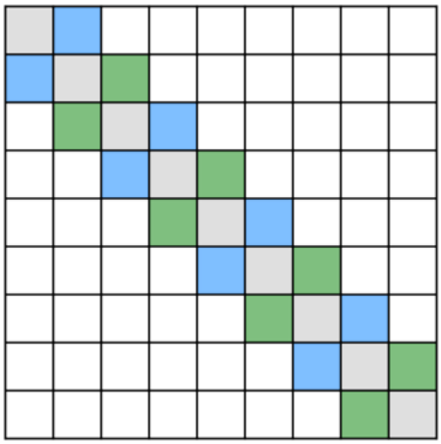
- maximum-pivot algorithm chooses the largest off-diagonal entry.
This tends to reduce the number of rotations,
but after each rotation the next largest pivot must be selected from the updated matrix.
Therefore the pivot rotations are processed one at a time.

- Cyclic Jacobi gives up largest-pivot selection and follows a predetermined schedule.
It tends to perform more rotations, but the schedule can be divided into nonintersecting batches,
and all rotations within a batch can be algebraically processed in parallel.

Householder tridiagonalization possibly gives this parallel strategy a favorable first batch:
all off-diagonal E is concentrated in the first off-diagonal band.

Define

- tridiagonal n × n matrix as T
- first off-diagonal band entries as $e_1 \ldots e_{n-1}$
- off-diagonal energy as $\Omega^2(T) = 2 \sum_{i=1}^{n-1} \left((e_i)^2 \right)$



We have two options for the first batch:

- Blue batch: (1,2), (3,4), (5,6), ...
- Green batch: (2,3), (4,5), (6,7), ...

No two pivots inside one batch share an index,
so all pivots of one color can be processed simultaneously.

- The off-diagonal E concentrated in the blue entries is

$$\Omega^2_{\text{blue}} = 2\left((e_1)^2 + (e_3)^2 + (e_5)^2 + \ldots\right)$$

- The off diagonal E concentrated in the green entries is

$$\Omega^2_{\text{green}} = 2\left((e_2)^2 + (e_4)^2 + (e_6)^2 + \ldots\right)$$

- The total off diagonal E is

$$\Omega^2(T) = \Omega^2_{\text{blue}} + \Omega^2_{\text{green}}.$$

After this first batch, fill-in begins,
and the Jacobi algorithm proceeds as usual.

Fill in with the first Jacobi batch
after tridiagonalization

T is a 4×4 tridiagonal matrix, and we select the green pivot pair (2,3).

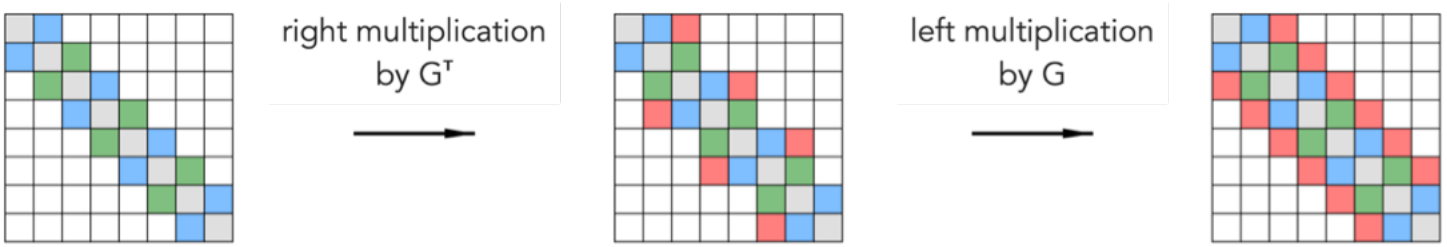
$$T = \begin{bmatrix} d_1 & e_1 & 0 & 0 \\ e_1 & d_2 & e_2 & 0 \\ 0 & e_2 & d_3 & e_3 \\ 0 & 0 & e_3 & d_4 \end{bmatrix} \qquad G = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c & s & 0 \\ 0 & -s & c & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T\,G^T = \begin{bmatrix} d_1 & c\,e_1 & -s\,e_1 & 0 \\ e_1 & * & * & 0 \\ 0 & * & * & e_3 \\ 0 & s\,e_3 & c\,e_3 & d_4 \end{bmatrix} \quad \text{and} \quad A' = G\,(T\,G^T) = \begin{bmatrix} d_1 & c\,e_1 & -s\,e_1 & 0 \\ c\,e_1 & d_2' & 0 & s\,e_3 \\ -s\,e_1 & 0 & d_3' & c\,e_3 \\ 0 & s\,e_3 & c\,e_3 & d_4 \end{bmatrix}$$

- Right-multiplication by G^T fills 2 entries in the columns of the pivots, as shown
- Left-multiplication by G fills 2 entries in the rows of the pivots

- The fill-in advances exactly one diagonal farther out

The same local rule repeats for the green batch in a larger matrix



Color coding:
- Green = selected pivots
- Blue = neighboring first-band entries
- Red = new fill-in

Experiment setup and data collection

Ⓐ
Implementation and hardware

Two CUDA C++ programs were used for the numerical experiments. Both used NVIDIA cuSOLVER; the eigenvector-validation program also used cuBLAS.

The computations were performed off-site on a rented remote GPU instance.

GPU: NVIDIA GeForce RTX 3060
- Ampere architecture
- compute capability 8.6
- 3584 CUDA cores
- approximately 11.63 GiB visible global GPU memory

The number of CUDA cores describes the available GPU hardware. cuSOLVER controls how computations are scheduled across those cores; it does not expose a one-to-one correspondence between CUDA cores and simultaneous Jacobi rotations.

Ⓑ

Runtime and sweep-count experiment

The first program compared two eigenvalue-only paths.

① Standalone Jacobi

A
↓
cuSOLVER cusolverDnDsyevj
↓
eigenvalues

② Householder preprocessing + Jacobi

A
↓
cuSOLVER cusolverDnDsytrd
↓
tridiagonal T
↓
cuSOLVER cusolverDnDsyevj
↓
eigenvalues

Both paths used

• double precision

- eigenvalues only
- Jacobi relative tolerance = 10^{-12}
- maximum Jacobi sweeps = 100
- exactly the same input matrix A

Matrix generation and host-to-device transfer were excluded from timing.
For the preprocessing path, Householder time and Jacobi time were measured separately,
as well as their combined runtime.

The program recorded runtimes, Jacobi sweep counts, residuals, and eigenvalues.
The two eigenvalue sets were sorted and compared.

©

Eigenvector validation and energy-distribution experiment

A second, untimed program examined the Householder + Jacobi path in more detail.

Before tridiagonalization it measured

- diagonal Frobenius energy of A
- off-diagonal Frobenius energy of A

After cuSOLVER cusolverDnDsytrd it measured

- diagonal Frobenius energy of T
- first-off-diagonal-band Frobenius energy of T
- preservation of total Frobenius energy

For eigenvector validation, the Householder reflectors produced by cusolverDnDsytrd
were converted to the orthogonal matrix Q using cusolverDnDorgtr.

Jacobi was then run on T with eigenvectors enabled, producing Z. The eigenvectors of the original matrix were reconstructed as

$$V = Q\,Z.$$

The program then measured

- $$\frac{\| V^{T} V - I \|_{F}}{\| I \|_{F}}$$
- $\max | V^{T} V - I |$

This experiment was not used for runtime comparisons.

Ⓓ

Common test setup

Both programs used the same 6 matrix families, matrix sizes, numbers of independently generated matrices, and generation rules described below.

They also used the same base random seed.

Every generated matrix was scaled by one scalar so that

$$\| A \|_{F} = \sqrt{n}\,.$$

Thus the 6 matrix families were tested at the same RMS eigenvalue scale.

Ⓔ

Data handling

The two programs printed their measured results to the console.

ChatGPT was used to transcribe and organize those console values into the data arrays used to construct the following tables.

The measurements themselves were produced by the CUDA programs. Derived quantities, including runtime and sweep-count percentage changes and changes in energy percentages, are calculated from the stored measurements rather than entered manually.

Symmetric matrices used for testing

Matrix family	Description	Construction	Typical setting
General dense	Dense with independent and evenly distributed entries	Independent entries are sampled from U[−1, 1] and mirrored across the diagonal	
Dense clustered spectrum	Dense with eigenvalues concentrated in four narrow clusters	• Eigenvalues are generated in four clusters centered at −1.5, −0.5, 0.5, and 1.5, with independent spread in $[-10^{-3}, 10^{-3}]$. • The resultant diagonal matrix is then similarity-transformed by a product of randomized orthogonal plane rotations	Closely spaced vibration modes or energy levels
Strictly diagonally dominant	Diagonal entry dominates every row	• Random symmetric off-diagonal entries are generated from U[−1, 1]. • For each row, the diagonal entry is set to one plus the sum of the absolute values of all off-diagonal entries in that row.	Differential-equation, circuit, and network models
Weakly coupled dense blocks	• Two dense diagonal blocks • Two weakly coupled off-diagonal blocks	• Entries inside the diagonal blocks are sampled from U[−1, 1] • Entries in the off-diagonal blocks are sampled at 0.05 times that scale. • The blocks are mirrored	Strongly interacting subsystems with weak coupling between them (such as networks)
Dense correlation	Dense positive-definite matrix with equal diagonal entries	• A dense positive-definite matrix is generated by orthogonal similarity transformations. • Rows and columns are then rescaled so that all diagonal entries are 1.	Standardized multivariate data and correlation analysis
Unequal-variance covariance	Dense positive-definite matrix with large diagonal spread	• Start with a dense correlation matrix. • For each variable, sample the logarithm of its standard deviation from 0 to 1. • This gives standard deviations from 1 to 10 and variances from 1 to 100 before normalization.	Adverse covariance stress test: unstandardized variables with strongly unequal scales

The first four matrix groups represent distinct structures that occur in practical eigenvalue problems.

The last two groups were suggested directly by the preceding energy analysis.

Diagonal E increases when the difference (a − d) lies between the two boundary values

$$\min\left(\frac{2\,x\,b}{y}\,,\quad\frac{-2\,y\,b}{x}\right) < (a-d) < \max\left(\frac{2\,x\,b}{y}\,,\quad\frac{-2\,y\,b}{x}\right)$$

Dense correlation matrices with $A_{ii} = 1$ are an algebraically favorable and common eigenvalue problem.

They are obtained by centering the data and standardizing each variable to unit variance.
Experimentally, they proved approximately as favorable as the general dense matrices.

At the opposite end, consider two centered variables $\vec{u}$ and $\vec{v}$, and scale the second variable by a factor s:

$$X = \left[\begin{array}{c|c} \vec{u} & s\,\vec{v} \end{array}\right] \qquad X^{T}X = \left[\begin{array}{c|c} |\,\vec{u}\,|^2 & s\,\vec{u}^{T}\vec{v} \\ \hline s\,\vec{u}^{T}\vec{v} & s^2\,|\,\vec{v}\,|^2 \end{array}\right]$$

As s increases, one diagonal entry grows as s^2 while the off-diagonal coupling grows only as s.
The matrix therefore moves toward the unfavorable condition $|\,a - d\,| \gg |\,b\,|$.

This is an adverse rather than a typical data-analysis case:
variables that differ greatly in scale are usually standardized first, producing a correlation matrix.

We retain the unstandardized covariance case as a deliberate stress test.
Experimentally, Householder transferred substantial energy off the diagonal, and the subsequent Jacobi computation required more sweeps and longer runtimes.

The diagonal spread is inexpensive to measure in advance and can warn that

preprocessing may be unfavorable.

Results: sweep counts by matrix size and type

General dense

	16×16 20 matrices	32×32 20 matrices	64×64 20 matrices	128×128 10 matrices	256×256 10 matrices	512×512 6 matrices	1024×1024 5 matrices
Jacobi after Householder	5.55	6.10	5.10	6.00	7.00	7.50	8.00
Jacobi	6.00	7.00	6.35	7.90	9.00	10.17	11.00
$100.0 \times \frac{\text{JacobiStandalone - JacobiAfterHouseholder}}{\text{JacobiStandalone}}$	7.5%	12.9%	19.7%	24.1%	22.2%	26.2%	27.3%

Dense clustered spectrum

	16×16 20 matrices	32×32 20 matrices	64×64 20 matrices	128×128 10 matrices	256×256 10 matrices	512×512 6 matrices	1024×1024 5 matrices
Jacobi after Householder	5.25	6.40	5.95	6.60	7.00	7.83	8.80
Jacobi	7.20	8.95	8.95	11.00	13.10	15.00	16.00
$100.0 \times \frac{\text{JacobiStandalone - JacobiAfterHouseholder}}{\text{JacobiStandalone}}$	27.1%	28.5%	33.5%	40.0%	46.6%	47.8%	45.0%

Strictly diagonally dominant

	16×16 20 matrices	32×32 20 matrices	64×64 20 matrices	128×128 10 matrices	256×256 10 matrices	512×512 6 matrices	1024×1024 5 matrices
Jacobi after Householder	5.50	6.05	5.10	6.00	7.00	7.17	8.00
Jacobi	6.00	6.85	6.05	7.40	8.90	10.00	11.00
$100.0 \times \frac{\text{JacobiStandalone - JacobiAfterHouseholder}}{\text{JacobiStandalone}}$	8.3%	11.7%	15.7%	18.9%	21.3%	28.3%	27.3%

Weakly coupled dense blocks

	16×16 20 matrices	32×32 20 matrices	64×64 20 matrices	128×128 10 matrices	256×256 10 matrices	512×512 6 matrices	1024×1024 5 matrices
Jacobi after Householder	5.25	6.10	5.30	6.00	7.00	7.50	8.00
Jacobi	5.95	7.00	6.05	7.50	9.00	10.00	11.00
$100.0 \times \frac{\text{JacobiStandalone - JacobiAfterHouseholder}}{\text{JacobiStandalone}}$	11.8%	12.9%	12.4%	20.0%	22.2%	25.0%	27.3%

Dense correlation

	16×16 20 matrices	32×32 20 matrices	64×64 20 matrices	128×128 10 matrices	256×256 10 matrices	512×512 6 matrices	1024×1024 5 matrices
Jacobi after Householder	5.25	6.05	5.15	6.00	7.00	7.17	8.00
Jacobi	5.95	7.00	6.70	8.00	9.00	10.00	11.00
$100.0 \times \frac{\text{JacobiStandalone - JacobiAfterHouseholder}}{\text{JacobiStandalone}}$	11.8%	13.6%	23.1%	25.0%	22.2%	28.3%	27.3%

Unequal-variance covariance

	16×16 20 matrices	32×32 20 matrices	64×64 20 matrices	128×128 10 matrices	256×256 10 matrices	512×512 6 matrices	1024×1024 5 matrices
Jacobi after Householder	5.50	6.55	5.70	6.20	7.20	8.00	9.00
Jacobi	4.80	5.05	4.95	5.50	6.00	7.00	8.00
$100.0 \times \frac{\text{JacobiStandalone - JacobiAfterHouseholder}}{\text{JacobiStandalone}}$	-14.6%	-29.7%	-15.2%	-12.7%	-20.0%	-14.3%	-12.5%

Results: runtimes by matrix size and type

General dense

	16×16 20 matrices	32×32 20 matrices	64×64 20 matrices	128×128 10 matrices	256×256 10 matrices	512×512 6 matrices	1024×1024 5 matrices
Jacobi after Householder	1.036 ms	1.988 ms	4.818 ms	12.478 ms	37.341 ms	114.762 ms	686.078 ms
Householder	0.145 ms	0.275 ms	0.823 ms	3.278 ms	5.430 ms	11.376 ms	34.098 ms
Householder + Jacobi	1.185 ms	2.268 ms	5.646 ms	15.761 ms	42.777 ms	126.150 ms	720.206 ms
Jacobi	1.139 ms	2.357 ms	6.124 ms	16.553 ms	51.285 ms	163.926 ms	997.764 ms
$100.0 \times \frac{\text{JacobiStandalone - Sum}}{\text{JacobiStandalone}}$	-4.0%	3.8%	7.8%	4.8%	16.6%	23.0%	27.8%

Dense clustered spectrum

	16×16 20 matrices	32×32 20 matrices	64×64 20 matrices	128×128 10 matrices	256×256 10 matrices	512×512 6 matrices	1024×1024 5 matrices
Jacobi after Householder	0.971 ms	1.905 ms	5.173 ms	13.609 ms	38.054 ms	117.732 ms	713.047 ms
Householder	0.145 ms	0.275 ms	0.823 ms	3.278 ms	5.430 ms	11.377 ms	34.205 ms
Householder + Jacobi	1.121 ms	2.185 ms	6.000 ms	16.892 ms	43.490 ms	129.121 ms	747.282 ms
Jacobi	1.349 ms	3.035 ms	8.397 ms	24.155 ms	76.436 ms	243.186 ms	1464.117 ms
$100.0 \times \frac{\text{JacobiStandalone - Sum}}{\text{JacobiStandalone}}$	16.9%	28.0%	28.5%	30.1%	43.1%	46.9%	49.0%

Strictly diagonally dominant

	16×16 20 matrices	32×32 20 matrices	64×64 20 matrices	128×128 10 matrices	256×256 10 matrices	512×512 6 matrices	1024×1024 5 matrices
Jacobi after Householder	0.971 ms	1.967 ms	4.750 ms	12.103 ms	36.496 ms	111.592 ms	676.081 ms
Householder	0.136 ms	0.275 ms	0.823 ms	3.278 ms	5.427 ms	11.382 ms	34.350 ms
Householder + Jacobi	1.111 ms	2.247 ms	5.577 ms	15.386 ms	41.928 ms	122.986 ms	710.461 ms
Jacobi	1.056 ms	2.287 ms	5.869 ms	15.972 ms	48.808 ms	155.674 ms	967.720 ms
$100.0 \times \frac{\text{JacobiStandalone - Sum}}{\text{JacobiStandalone}}$	-5.2%	1.7%	5.0%	3.7%	14.1%	21.0%	26.6%

Weakly coupled dense blocks

	16×16 20 matrices	32×32 20 matrices	64×64 20 matrices	128×128 10 matrices	256×256 10 matrices	512×512 6 matrices	1024×1024 5 matrices
Jacobi after Householder	0.906 ms	1.980 ms	4.861 ms	12.491 ms	37.202 ms	114.787 ms	698.173 ms
Householder	0.133 ms	0.275 ms	0.823 ms	3.278 ms	5.430 ms	11.378 ms	34.337 ms
Householder + Jacobi	1.043 ms	2.260 ms	5.688 ms	15.773 ms	42.638 ms	126.177 ms	732.539 ms
Jacobi	1.027 ms	2.356 ms	5.760 ms	16.292 ms	50.805 ms	161.687 ms	1003.881 ms
$100.0 \times \frac{\text{JacobiStandalone - Sum}}{\text{JacobiStandalone}}$	-1.6%	4.1%	1.2%	3.2%	16.1%	22.0%	27.0%

Dense correlation

	16×16 20 matrices	32×32 20 matrices	64×64 20 matrices	128×128 10 matrices	256×256 10 matrices	512×512 6 matrices	1024×1024 5 matrices
Jacobi after Householder	0.953 ms	1.996 ms	4.781 ms	12.628 ms	37.439 ms	116.905 ms	697.059 ms
Householder	0.139 ms	0.278 ms	0.830 ms	3.307 ms	5.478 ms	11.671 ms	34.610 ms
Householder + Jacobi	1.096 ms	2.278 ms	5.615 ms	15.939 ms	42.922 ms	128.587 ms	731.700 ms
Jacobi	1.090 ms	2.373 ms	6.268 ms	16.932 ms	52.048 ms	168.136 ms	1013.709 ms
$100.0 \times \frac{\text{JacobiStandalone - Sum}}{\text{JacobiStandalone}}$	-0.6%	4.0%	10.4%	5.9%	17.5%	23.5%	27.8%

Unequal-variance covariance

	16×16 20 matrices	32×32 20 matrices	64×64 20 matrices	128×128 10 matrices	256×256 10 matrices	512×512 6 matrices	1024×1024 5 matrices
Jacobi after Householder	0.960 ms	2.074 ms	5.124 ms	13.151 ms	39.312 ms	126.357 ms	770.029 ms
Householder	0.134 ms	0.278 ms	0.830 ms	3.305 ms	5.475 ms	11.658 ms	34.614 ms
Householder + Jacobi	1.097 ms	2.356 ms	5.959 ms	16.460 ms	44.793 ms	138.027 ms	804.672 ms
Jacobi	0.837 ms	1.710 ms	4.336 ms	11.630 ms	34.362 ms	110.546 ms	685.951 ms
$100.0 \times \frac{\text{JacobiStandalone - Sum}}{\text{JacobiStandalone}}$	-31.2%	-37.7%	-37.4%	-41.5%	-30.4%	-24.9%	-17.3%

Results: eigenvector orthogonality

General dense

	16×16 20 matrices	32×32 20 matrices	64×64 20 matrices	128×128 10 matrices	256×256 10 matrices	512×512 6 matrices	1024×1024 5 matrices
$\frac{\lVert V^{T} V - I \rVert_{F}}{\lVert I \rVert_{F}}$	2.164e-15	4.667e-15	1.242e-14	3.769e-14	1.041e-13	2.622e-13	6.283e-13
$\max \lvert V^{T} V - I \rvert$	3.630e-15	8.071e-15	2.115e-14	5.820e-14	1.612e-13	3.905e-13	8.666e-13

Dense clustered spectrum

	16×16 20 matrices	32×32 20 matrices	64×64 20 matrices	128×128 10 matrices	256×256 10 matrices	512×512 6 matrices	1024×1024 5 matrices
$\frac{\lVert V^{T} V - I \rVert_{F}}{\lVert I \rVert_{F}}$	3.113e-15	7.597e-15	2.575e-14	6.997e-14	1.561e-13	3.497e-13	7.460e-13
$\max \lvert V^{T} V - I \rvert$	4.652e-15	1.040e-14	3.209e-14	8.240e-14	1.766e-13	3.889e-13	8.246e-13

Strictly diagonally dominant

	16×16 20 matrices	32×32 20 matrices	64×64 20 matrices	128×128 10 matrices	256×256 10 matrices	512×512 6 matrices	1024×1024 5 matrices
$\frac{\lVert V^{T} V - I \rVert_{F}}{\lVert I \rVert_{F}}$	2.256e-15	4.601e-15	1.267e-14	3.700e-14	1.049e-13	2.614e-13	6.330e-13
$\max \lvert V^{T} V - I \rvert$	3.686e-15	7.649e-15	2.089e-14	5.900e-14	1.596e-13	3.747e-13	9.099e-13

Weakly coupled dense blocks

	16×16 20 matrices	32×32 20 matrices	64×64 20 matrices	128×128 10 matrices	256×256 10 matrices	512×512 6 matrices	1024×1024 5 matrices
$\frac{\lVert V^{T} V - I \rVert_{F}}{\lVert I \rVert_{F}}$	2.197e-15	4.944e-15	1.323e-14	3.813e-14	1.047e-13	2.649e-13	6.323e-13
$\max \lvert V^{T} V - I \rvert$	3.919e-15	8.482e-15	2.137e-14	6.208e-14	1.505e-13	4.011e-13	8.643e-13

Dense correlation

	16×16 20 matrices	32×32 20 matrices	64×64 20 matrices	128×128 10 matrices	256×256 10 matrices	512×512 6 matrices	1024×1024 5 matrices
$\frac{\lVert V^T V - I \rVert_F}{\lVert I \rVert_F}$	2.213e-15	4.821e-15	1.268e-14	3.860e-14	1.060e-13	2.656e-13	6.419e-13
$\max \lvert V^T V - I \rvert$	3.864e-15	8.493e-15	2.109e-14	6.091e-14	1.614e-13	3.718e-13	8.732e-13

Unequal-variance covariance

	16×16 20 matrices	32×32 20 matrices	64×64 20 matrices	128×128 10 matrices	256×256 10 matrices	512×512 6 matrices	1024×1024 5 matrices
$\frac{\lVert V^T V - I \rVert_F}{\lVert I \rVert_F}$	2.125e-15	5.402e-15	1.484e-14	4.312e-14	1.144e-13	2.857e-13	6.768e-13
$\max \lvert V^T V - I \rvert$	3.719e-15	8.704e-15	2.356e-14	6.821e-14	1.713e-13	4.193e-13	9.287e-13

Results: energy before and after tridiagonalization

We previously showed that Householder tridiagonalization can
either increase or decrease diagonal energy.
The following measurements check whether, in practice,
it produces a substantial unfavorable shift of energy away from the diagonal.

General dense

	16×16 20 matrices	32×32 20 matrices	64×64 20 matrices	128×128 10 matrices	256×256 10 matrices	512×512 6 matrices	1024×1024 5 matrices
$100 \times \frac{E\_\text{diagonal(A)}}{E(A)}$	6.785%	3.146%	1.488%	0.799%	0.390%	0.192%	0.097%
$100 \times \frac{E\_\text{off-diagonal(A)}}{E(A)}$	93.215%	96.854%	98.512%	99.201%	99.610%	99.808%	99.903%
$100 \times \frac{E\_\text{diagonal(T)}}{E(A)}$	9.699%	5.876%	2.855%	1.621%	0.818%	0.406%	0.190%
$100 \times \frac{E\_\text{first-band(T)}}{E(A)}$	90.301%	94.124%	97.145%	98.379%	99.182%	99.594%	99.810%
$100 \times \frac{E\_\text{diagonal(T)} - E\_\text{diagonal(A)}}{E(A)}$	2.914%	2.730%	1.367%	0.822%	0.427%	0.215%	0.093%

Dense clustered spectrum

	16×16 20 matrices	32×32 20 matrices	64×64 20 matrices	128×128 10 matrices	256×256 10 matrices	512×512 6 matrices	1024×1024 5 matrices
$100 \times \frac{E\_\text{diagonal(A)}}{E(A)}$	11.982%	6.342%	3.108%	1.641%	0.828%	0.394%	0.204%
$100 \times \frac{E\_\text{off-diagonal(A)}}{E(A)}$	88.018%	93.658%	96.892%	98.359%	99.172%	99.606%	99.796%
$100 \times \frac{E\_\text{diagonal(T)}}{E(A)}$	41.961%	43.139%	35.370%	35.641%	44.565%	45.489%	43.945%
$100 \times \frac{E\_\text{first-band(T)}}{E(A)}$	58.039%	56.861%	64.630%	64.359%	55.435%	54.511%	56.055%
$100 \times \frac{E\_\text{diagonal(T)} - E\_\text{diagonal(A)}}{E(A)}$	29.979%	36.797%	32.262%	34.000%	43.738%	45.095%	43.741%

Strictly diagonally dominant

	16×16 20 matrices	32×32 20 matrices	64×64 20 matrices	128×128 10 matrices	256×256 10 matrices	512×512 6 matrices	1024×1024 5 matrices
A matrix: $100 \times \frac{E\_\text{diagonal(A)}}{E\text{(A)}}$	93.621%	96.382%	98.064%	98.996%	99.489%	99.742%	99.870%
A matrix: $100 \times \frac{E\_\text{off-diagonal(A)}}{E\text{(A)}}$	6.379%	3.618%	1.936%	1.004%	0.511%	0.258%	0.130%
T matrix: $100 \times \frac{E\_\text{diagonal(T)}}{E\text{(A)}}$	93.119%	95.853%	97.708%	98.759%	99.368%	99.681%	99.838%
T matrix: $100 \times \frac{E\_\text{first-band(T)}}{E\text{(A)}}$	6.881%	4.147%	2.292%	1.241%	0.632%	0.319%	0.162%
$100 \times \frac{E\_\text{diagonal(T)} - E\_\text{diagonal(A)}}{E\text{(A)}}$	-0.502%	-0.529%	-0.356%	-0.237%	-0.121%	-0.061%	-0.033%

Weakly coupled dense blocks

	16×16 20 matrices	32×32 20 matrices	64×64 20 matrices	128×128 10 matrices	256×256 10 matrices	512×512 6 matrices	1024×1024 5 matrices
A matrix: $100 \times \frac{E\_\text{diagonal(A)}}{E\text{(A)}}$	13.988%	6.351%	3.058%	1.544%	0.781%	0.400%	0.195%
A matrix: $100 \times \frac{E\_\text{off-diagonal(A)}}{E\text{(A)}}$	86.012%	93.649%	96.942%	98.456%	99.219%	99.600%	99.805%
T matrix: $100 \times \frac{E\_\text{diagonal(T)}}{E\text{(A)}}$	19.400%	9.885%	4.643%	2.265%	1.032%	0.469%	0.215%
T matrix: $100 \times \frac{E\_\text{first-band(T)}}{E\text{(A)}}$	80.600%	90.115%	95.357%	97.735%	98.968%	99.531%	99.785%
$100 \times \frac{E\_\text{diagonal(T)} - E\_\text{diagonal(A)}}{E\text{(A)}}$	5.412%	3.534%	1.585%	0.721%	0.251%	0.069%	0.020%

Dense correlation

	16×16 20 matrices	32×32 20 matrices	64×64 20 matrices	128×128 10 matrices	256×256 10 matrices	512×512 6 matrices	1024×1024 5 matrices
A matrix: $100 \times \frac{E\_\text{diagonal(A)}}{E\text{(A)}}$	92.664%	92.711%	92.655%	92.633%	92.295%	92.284%	92.304%
A matrix: $100 \times \frac{E\_\text{off-diagonal(A)}}{E\text{(A)}}$	7.336%	7.289%	7.345%	7.367%	7.705%	7.716%	7.696%
T matrix: $100 \times \frac{E\_\text{diagonal(T)}}{E\text{(A)}}$	93.762%	93.397%	93.131%	92.909%	92.480%	92.374%	92.363%
T matrix: $100 \times \frac{E\_\text{first-band(T)}}{E\text{(A)}}$	6.238%	6.603%	6.869%	7.091%	7.520%	7.626%	7.637%
$100 \times \frac{E\_\text{diagonal(T)} - E\_\text{diagonal(A)}}{E\text{(A)}}$	1.098%	0.686%	0.476%	0.276%	0.186%	0.089%	0.059%

Unequal-variance covariance

	16×16 20 matrices	32×32 20 matrices	64×64 20 matrices	128×128 10 matrices	256×256 10 matrices	512×512 6 matrices	1024×1024 5 matrices
A matrix: $100 \times \frac{E\_\text{diagonal(A)}}{E\text{(A)}}$	96.863%	96.614%	96.801%	96.845%	96.649%	96.614%	96.562%
A matrix: $100 \times \frac{E\_\text{off-diagonal(A)}}{E\text{(A)}}$	3.137%	3.386%	3.199%	3.155%	3.351%	3.386%	3.438%
T matrix: $100 \times \frac{E\_\text{diagonal(T)}}{E\text{(A)}}$	71.399%	69.502%	68.959%	68.111%	68.067%	67.956%	67.782%
T matrix: $100 \times \frac{E\_\text{first-band(T)}}{E\text{(A)}}$	28.601%	30.498%	31.041%	31.889%	31.933%	32.044%	32.218%
$100 \times \frac{E\_\text{diagonal(T)} - E\_\text{diagonal(A)}}{E\text{(A)}}$	-25.464%	-27.112%	-27.842%	-28.734%	-28.582%	-28.658%	-28.780%

Conclusions

Ⓐ
Algebraic conclusions

Householder tridiagonalization concentrates all off-diagonal entries into the first off-diagonal band.

Jacobi rotations do not preserve those zeros: fill-in begins immediately. Therefore the tridiagonal structure itself is temporary.

However, rotations acting on nonintersecting pivot pairs can be assembled into a Jacobi batch. For a tridiagonal matrix, the two alternating first-band batches partition all off-diagonal energy, so at least one contains at least half of it.

Householder similarity can also redistribute energy between the diagonal and off-diagonal parts of the matrix. The local algebra identifies two regimes: similar diagonal entries are favorable, while a large diagonal difference relative to the existing off-diagonal coupling can be unfavorable.

Thus concentration of off-diagonal energy is guaranteed, but favorable energy redistribution is not.

Ⓑ
Experimental conclusions

For the tested NVIDIA cuSOLVER cusolverDnDsyevj implementation, Householder preprocessing reduced the measured Jacobi sweep count for 5 of the 6 tested matrix families.

For sufficiently large matrices in these favorable families, the reduction in Jacobi runtime exceeded the additional cost of tridiagonalization, so the complete Householder + cusolverDnDsyevj path was faster.

Dense correlation matrices provided an important favorable case: their equal diagonal entries match the favorable algebraic regime, and experimentally they performed approximately as well as general dense matrices.

The unequal-variance covariance family was constructed as an adverse test case. Its large diagonal spread produced substantial transfer of energy off the diagonal, and the subsequent Jacobi computation required more sweeps and longer runtime.

The reconstructed eigenvectors remained orthogonal to approximately the level expected from floating-point roundoff.

©

Scope of the result

The algebra establishes a structural advantage of the tridiagonal starting matrix: all off-diagonal energy is concentrated in the first band, where one of the two alternating nonintersecting batches contains at least half of it.

It also establishes an important limitation: Householder similarity can move energy either toward or away from the diagonal. The benefit therefore depends on the matrix,
not merely on the concentration produced by tridiagonalization.

The experiments reproduce both algebraic regimes. In five tested families, Householder preprocessing reduced the subsequent Jacobi work. In the deliberately
adverse unequal-variance covariance family, it increased that work.

The algebra does not determine whether Jacobi savings will recover the $O(n^3)$ cost of Householder tridiagonalization. That question remains implementation- and matrix-dependent and is measured here for cusolverDnDsyevj.

The central conclusion is therefore conditional rather than universal:

Householder tridiagonalization can substantially improve a parallel Jacobi eigensolver,
but its effect depends on how the similarity transformation redistributes energy. Matrices with nearly equal diagonal entries are favorable candidates, while large diagonal spread provides an inexpensive warning that preprocessing may be unfavorable.

References

① L. N. Trefethen and D. Bau III
Numerical Linear Algebra
SIAM, 2022.

② F. J. Corbató
On the Coding of Jacobi's Method
for Computing Eigenvalues and Eigenvectors of Real Symmetric Matrices
Journal of the ACM, 10(2), 123–125, 1963.

③ NVIDIA
cuSOLVER Library, CUDA Toolkit Documentation
Documentation for cusolverDnDsyevj, cusolverDnDsytrd, and cusolverDnDorgtr.