Akademia Górniczo-Hutnicza
im. Stanisława Staszica w Krakowie

AGH University of Krakow

Numerical Methods in Physics

Project 03
Diagonalization of Symmetric Tridiagonal Matrices
Quantum Harmonic Oscillator

Hamzeh Khanpour

hamzeh.khanpour@fis.agh.edu.pl

Faculty of Physics and Applied Computer Science
AGH University of Krakow

Why this laboratory?

Two questions

How do we solve a symmetric tridiagonal eigenvalue problem in C++?

How do we verify the computed eigenvalues and eigenvectors by comparing them with the analytical quantum harmonic oscillator solution?

Scientific workflow

Model $\longrightarrow$ discretize $\longrightarrow$ diagonalize $\longrightarrow$ verify $\longrightarrow$ interpret.

Course sources

S01: Tomasz Chwiej, Numerical Methods in Physics course page.
S02: Project 03 handout: Diagonalization of symmetric tridiagonal matrix (quantum harmonic oscillator problem).
S03: Lecture 3: Diagonalization of dense matrices.

Technical sources

Netlib LAPACKE: C interface to LAPACK.
LAPACK routine: `DSTEV` (eigenvalues and eigenvectors of a real symmetric tridiagonal matrix).

From classical energy to the Hamiltonian operator

A physical question

In mechanics, the total energy of a particle is the sum of *kinetic energy* and *potential energy*:

$$E = T + V.$$

In quantum mechanics, the state of a particle is described by a *wavefunction* $\Psi(\mathbf{r}, t)$, and measurable quantities are represented by operators.

The energy operator

The operator corresponding to the total energy is the Hamiltonian:

$$\hat{H} = \hat{T} + \hat{V}.$$

For a single particle in a potential $V(\mathbf{r})$,

$$\hat{H}(\mathbf{r}) = -\frac{\hbar^2}{2m}\nabla^2 + V(\mathbf{r}).$$

Here the Laplacian ∇^2 represents the kinetic-energy part, while $V(\mathbf{r})$ represents the interaction or confining potential.

From the Schrödinger equation to stationary states

Time-dependent Schrödinger equation

The evolution of a quantum state is governed by $i\hbar \frac{\partial}{\partial t} \Psi(\mathbf{r}, t) = \hat{H} \Psi(\mathbf{r}, t)$. This is the fundamental equation of nonrelativistic quantum mechanics.

Stationary states

For many important systems, we look for solutions with a simple time dependence,

$$\Psi(\mathbf{r}, t) = \psi(\mathbf{r})\, e^{-iEt/\hbar},$$

where E is a constant energy and $\psi(\mathbf{r})$ describes the spatial part. Substituting this form into the time-dependent equation gives

$$\hat{H}\psi(\mathbf{r}) = E\psi(\mathbf{r}).$$

This is the time-independent Schrödinger equation.

Interpretation

This equation is an *eigenvalue problem*: $\hat{H}\psi = E\psi$. The eigenvalues E are the allowed energy levels, and the eigenfunctions ψ are the corresponding stationary quantum states.

The model used in this laboratory

In this project we study a particle moving in the parabolic potential

$$V(x) = \frac{x^2}{2}.$$

Using the dimensionless units adopted in the handout ($\hbar = 1$, $m = 1$, $\omega = 1$), the Hamiltonian becomes

$$\hat{H}(x) = -\frac{1}{2}\frac{d^2}{dx^2} + \frac{x^2}{2}, \qquad x \in (-\infty, \infty).$$

Its stationary Schrödinger equation

The wavefunction $\psi_k(x)$ and the corresponding energy E_k satisfy

$$\hat{H}\psi_k(x) = E_k\psi_k(x),$$

that is,

$$-\frac{1}{2}\frac{d^2\psi_k(x)}{dx^2} + \frac{x^2}{2}\psi_k(x) = E_k\psi_k(x).$$

So our physics problem is already an eigenvalue problem in continuous form.

Why diagonalization matters in physics

Two physical questions

Which vibrational frequencies are allowed? Which energy states can a quantum system occupy? A model becomes an operator, then a matrix, and finally an eigenvalue problem.

$$\underbrace{\hat{H}\psi = E\psi}_{\text{continuous physics}} \longrightarrow \underbrace{H\mathbf{v} = E\mathbf{v}}_{\text{computable matrix}} .$$

What the computer returns

Eigenvalues tell us the allowed energies; eigenvectors describe sampled spatial states. This laboratory studies a one-dimensional particle bound by a parabolic potential.

What you learn and what you implement

AGH

www.agh.edu.pl

We teach and demonstrate

- Eigenpairs, symmetry and matrix storage.
- Finite differences and the Hamiltonian.
- LAPACKE, normalization and Hermite recurrence.
- Small, complete E01–E09 toy programs.

You implement independently

- The assigned oscillator matrix and solver.
- Your numerical and analytical outputs.
- Your own plots and numerical interpretation.
- The home report, including the finer grid.

How the 135-minute laboratory works

Time (minutes)	Activity
0–10	Purpose, files and computing readiness
10–40	Board derivations and selected E01–E09 examples
40–50	Official project briefing and outputs
50–120	**70 minutes: write and test your own code**
120–135	Diagnose results; explain report work

Use the PDF while coding

Not every reference frame is spoken through during the lab. Revisit the derivations, compiler commands and independent checks when you need them.

Student package: where to start

File map

`examples/` - ten independent C++ sources
`bin/` - compiled executables
`data/`, `plots/` - your output files
`plotting/` - CSV plotting example
`tests/`, `Makefile`, `README.md`

Terminal preflight

```
pwd
ls examples
g++ -version
make help
make standard
make check-standard
```

Every .cpp file has its own main(). Do not link them all together.

The LAPACKE examples require the real header, linker library and runtime dependencies.

From Lab 2 to Lab 3: a new question

	Previous: LU	**Today: diagonalization**
Equation	$A\mathbf{x} = \mathbf{b}$	$A\mathbf{v} = \lambda\mathbf{v}$
Given	Matrix and source	Matrix (Hamiltonian)
Find	Unknown solution	Eigenvalues and directions
Check	$\mathbf{b} - A\mathbf{x}$	$A\mathbf{v} - \lambda\mathbf{v}$

What does an eigenvector mean?

It is a nonzero direction that the matrix does not turn into a different direction: it is only multiplied by the eigenvalue λ.

Read every symbol in an eigenvalue equation

$$A\mathbf{v} = \lambda\mathbf{v}, \qquad \underbrace{A}_{n\times n}\underbrace{\mathbf{v}}_{n\times 1} = \underbrace{\lambda}_{\text{scalar}}\underbrace{\mathbf{v}}_{n\times 1}, \quad \mathbf{v} \neq \mathbf{0}.$$

Component equation

For each matrix row i, $\sum_{j=0}^{n-1} A_{ij} v_j = \lambda v_i$. The C++ component `v[0]` is the mathematical v_0.

Why exclude the zero vector?

$A\mathbf{0} = \lambda\mathbf{0}$ holds for every λ; it contains no spectral information.

A symmetric 2-by-2 eigenproblem: eigenvalues

$$A = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}, \qquad \det(A - \lambda I) = 0.$$

$$A - \lambda I = \begin{pmatrix} 2-\lambda & 1 \\ 1 & 2-\lambda \end{pmatrix},$$

$$\begin{aligned} \det(A - \lambda I) &= (2-\lambda)^2 - 1 \\ &= \lambda^2 - 4\lambda + 3 = (\lambda - 1)(\lambda - 3). \end{aligned}$$

Two roots

$\boxed{\lambda_1 = 1, \qquad \lambda_2 = 3}$. For large matrices we use numerical libraries, not an explicit characteristic polynomial.

The same 2-by-2: vectors and direct checks



$\lambda_1 = 1$

$(A - I)\mathbf{v} = 0$ gives $v_1 = -v_2$.

$$\mathbf{q}_1 = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 \\ -1 \end{pmatrix}.$$

$A\mathbf{q}_1 = \mathbf{q}_1$.

$\lambda_2 = 3$

$(A - 3I)\mathbf{v} = 0$ gives $v_1 = v_2$.

$$\mathbf{q}_2 = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

$A\mathbf{q}_2 = 3\mathbf{q}_2$.

Check both properties

$\mathbf{q}_1^T\mathbf{q}_1 = \mathbf{q}_2^T\mathbf{q}_2 = 1$; $\mathbf{q}_1^T\mathbf{q}_2 = (1 - 1)/2 = 0$.

E01: a complete tiny eigensystem in C++

From hand solution to verified code

`01_small_symmetric_eigenproblem.cpp` stores the two known normalized vectors, multiplies by A, and tests $A\mathbf{v}_k - \lambda_k \mathbf{v}_k$.

```
const std::array<double,2> eigenvalues{{1,3}};
const double Av=A[i][0]*vectors[k][0]
             +A[i][1]*vectors[k][1];
max_residual=std::max(max_residual,
             std::abs(Av-eigenvalues[k]*vectors[k][i]));
```

```
make bin/01_small_symmetric_eigenproblem
./bin/01_small_symmetric_eigenproblem
```

Expected check

$\lambda = (1, 3)$, vector overlap 0, residual approximately 0; the file prints `PASS E01` only after checking.

Symmetry gives orthonormal eigenvectors

$$Q = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix}, \qquad \Lambda = \begin{pmatrix} 1 & 0 \\ 0 & 3 \end{pmatrix}.$$

$$Q^T Q = I, \qquad A = Q \Lambda Q^T.$$

Reconstruct the original matrix

$$Q \Lambda Q^T = \tfrac{1}{2} \begin{pmatrix} 1 & 3 \\ -1 & 3 \end{pmatrix} \begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}.$$

The eigenvectors form the columns of Q; the corresponding energies form Λ.

Eigenvector signs and eigenvalue ordering

$$A\mathbf{v} = \lambda\mathbf{v} \quad \Longrightarrow \quad A(-\mathbf{v}) = \lambda(-\mathbf{v}).$$

What is arbitrary?

For a real, isolated eigenvalue, an eigensolver may return either $\mathbf{v}$ or $-\mathbf{v}$. The physical state is unchanged; it is not a numerical error.

What must stay paired?

If sorted eigenvalues are $\lambda_0 \leq \lambda_1 \leq \ldots$, eigenvector *column* k corresponds to eigenvalue λ_k. An arbitrary sign does not mean arbitrary state ordering.

Degenerate eigenvalues can admit arbitrary orthogonal combinations within the same subspace.

Recognize a symmetric tridiagonal matrix

$$T = \begin{pmatrix} d_0 & e_0 & 0 & 0 \\ e_0 & d_1 & e_1 & 0 \\ 0 & e_1 & d_2 & e_2 \\ 0 & 0 & e_2 & d_3 \end{pmatrix}.$$

Only the central band can be nonzero

$T_{ij} = 0$ when $|i - j| > 1$; $T_{i,i+1} = T_{i+1,i}$. Store just d and e.

One matrix, two short arrays

$$\underbrace{d = [2, 2, 2, 2]}_{n = 4 \text{ diagonal entries}} , \qquad \underbrace{e = [-1, -1, -1]}_{n - 1 = 3 \text{ offdiagonal entries}} .$$

For $\mathbf{v} = (1, 2, 3, 4)^T$

$(T\mathbf{v})_0 = d_0 v_0 + e_0 v_1 = 2 - 2 = 0,$
$(T\mathbf{v})_1 = e_0 v_0 + d_1 v_1 + e_1 v_2 = -1 + 4 - 3 = 0,$
$(T\mathbf{v})_3 = e_2 v_2 + d_3 v_3 = -3 + 8 = 5.$

$$\boxed{T\mathbf{v} = (0, 0, 0, 5)^T}.$$

No zeros outside the band need to be stored.

E02: compute a tridiagonal matrix–vector product

```
for (int i=0;i<4;++i) {
    y[i]=d[i]*v[i];
    if (i>0) y[i]+=e[i-1]*v[i-1];
    if (i+1<4) y[i]+=e[i]*v[i+1];
}
```

Run the full independent demonstration

```
make bin/02_tridiagonal_storage_matvec
./bin/02_tridiagonal_storage_matvec
```

Predict before running

$T\mathbf{v} = (0, 0, 0, 5)^T$; the source compares every component with that independently calculated answer.

The stationary Schr"odinger eigenproblem

$$\hat{H}\psi_k(x) = E_k\psi_k(x), \qquad \hat{H} = -\frac{1}{2}\frac{d^2}{dx^2} + V(x).$$

Kinetic part

$-\frac{1}{2}\, d^2\psi/dx^2$ describes the curvature contribution.

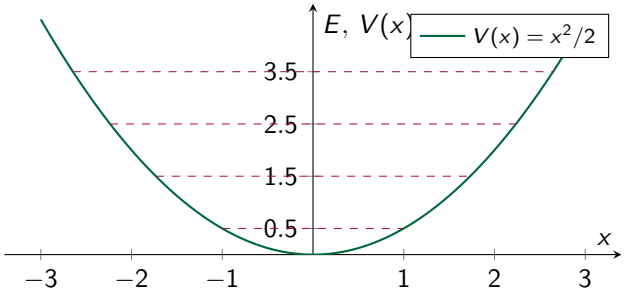
Potential part

$V(x)\psi(x)$ depends on position and confines the particle.

$\boxed{V(x) = \frac{1}{2}x^2}$ (dimensionless oscillator units: $\hbar = m = \omega = 1$).

Solving the eigenproblem gives the energies E_k and stationary wavefunctions ψ_k.

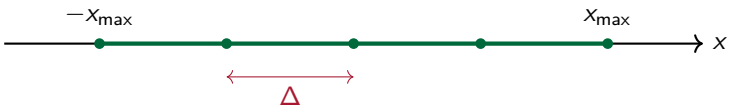
Parabolic confinement and energy levels



Dashed segments indicate *schematic level markers* at exact oscillator energies $E_k = k + \frac{1}{2}$; this is **not** numerical project output.

Infinite physical domain, finite computational box

$$x \in (-\infty, \infty) \quad \longrightarrow \quad [-x_{\max}, x_{\max}], \qquad \Delta = \frac{2x_{\max}}{n-1}.$$



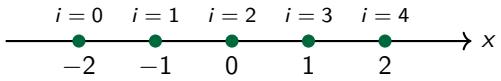
Two approximations to remember

We replace an infinite interval by a finite one, then replace derivatives by differences. Increasing n at fixed $x_{\max}$ reduces grid spacing; it does not enlarge the box.

Hand-built grid: five endpoint-inclusive nodes

$$n = 5, \quad x_{\max} = 2, \quad \Delta = \frac{2(2)}{5-1} = 1, \qquad x_i = -2 + i.$$

Index i	0	1	2	3	4
Position x_i	-2	-1	0	1	2

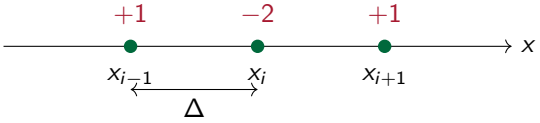


Derive the three-point second derivative

$$\psi(x_i + \Delta) = \psi_i + \Delta\psi_i' + \frac{\Delta^2}{2}\psi_i'' + \frac{\Delta^3}{6}\psi_i^{(3)} + O(\Delta^4),$$

$$\psi(x_i - \Delta) = \psi_i - \Delta\psi_i' + \frac{\Delta^2}{2}\psi_i'' - \frac{\Delta^3}{6}\psi_i^{(3)} + O(\Delta^4).$$

$$\boxed{\psi''(x_i) = \frac{\psi_{i-1} - 2\psi_i + \psi_{i+1}}{\Delta^2} + O(\Delta^2)}.$$



Odd Taylor terms cancel when the two expansions are added; after division the leading error is second order.

Finite differences: an error you can calculate

$$f(x) = x^4, \quad f''(x) = 12x^2, \qquad f''(1) = 12.$$

$$\begin{aligned} D_2 f(1;h) &= \frac{(1+h)^4 - 2 + (1-h)^4}{h^2} \\ &= \frac{12h^2 + 2h^4}{h^2} = 12 + 2h^2. \end{aligned}$$

h	Difference approximation	Absolute error
$1/2$	12.5	0.5
$1/4$	12.125	0.125

Check the order

Halving h makes the error four times smaller: $(h/2)^2/h^2 = 1/4$.

E03: run the central-difference error experiment

```
static double f(double x) { return x*x*x*x; }
const double approximation=
   (f(1+h)-2*f(1)+f(1-h))/(h*h);
const double error=std::abs(approximation-12.0);
```

Compile and run the independent example

```
make bin/03_central_second_difference
./bin/03_central_second_difference
```

Expected output (verify numerically)

$h = 0.5, 0.25, 0.125$, errors $0.5, 0.125, 0.03125$; successive ratios near 4.

No Schrödinger equation is solved in this example.

Endpoint and ghost-node conventions



At an interior node

The stencil uses $\psi_{i-1}, \psi_i, \psi_{i+1}$, all represented on the grid.

At an endpoint in the handout's $n \times n$ matrix

The same algebra would require a neighbor outside the stored grid. Writing only the assigned tridiagonal entries corresponds to $\psi_{-1} = 0$ and $\psi_n = 0$ as *external ghost values*.

Not the same as setting represented endpoints to zero

ψ_0 and ψ_{n-1} remain unknown matrix components. An interior-node Dirichlet formulation has a different Δ rule and must not replace the assignment.

From curvature to a kinetic-energy matrix

$$-\frac{1}{2}\psi''(x_i) \approx -\frac{1}{2}\frac{\psi_{i-1} - 2\psi_i + \psi_{i+1}}{\Delta^2}$$
$$= -\frac{1}{2\Delta^2}\psi_{i-1} + \frac{1}{\Delta^2}\psi_i - \frac{1}{2\Delta^2}\psi_{i+1}.$$

Read the matrix coefficients

The central site has kinetic coefficient $1/\Delta^2$; each immediate neighbor has coefficient $-1/(2\Delta^2)$; all farther neighbors have zero coefficients.

$$\boxed{K_{ii} = \Delta^{-2}}, \qquad \boxed{K_{i,i\pm1} = -\tfrac{1}{2\Delta^2}}.$$

The potential changes the diagonal only

$$\underbrace{-\tfrac{1}{2}\psi''(x_i)}_{\text{neighbor coupling}} + \underbrace{V(x_i)\psi_i}_{\text{same grid node}} = E\psi_i, \quad V(x_i) = \tfrac{1}{2}x_i^2.$$

$$(H\psi)_i = -\frac{1}{2\Delta^2}\psi_{i-1} + \left(\frac{1}{\Delta^2} + \frac{x_i^2}{2}\right)\psi_i - \frac{1}{2\Delta^2}\psi_{i+1}.$$

The assigned nonzero elements

$$\boxed{H_{ii} = \Delta^{-2} + \tfrac{1}{2}x_i^2}, \qquad \boxed{H_{i,i+1} = H_{i+1,i} = -1/(2\Delta^2)}.$$

The complete oscillator matrix: a five-node HAND example

$$n = 5,\ x_{\max} = 2,\ \Delta = 1,\ x = (-2, -1, 0, 1, 2).$$

$$d_i = 1 + \tfrac{1}{2}x_i^2 \ \Rightarrow\ d = (3, 1.5, 1, 1.5, 3), \qquad e_i = -\tfrac{1}{2}.$$

$$H = \begin{pmatrix} 3 & -\frac{1}{2} & 0 & 0 & 0 \\ -\frac{1}{2} & \frac{3}{2} & -\frac{1}{2} & 0 & 0 \\ 0 & -\frac{1}{2} & 1 & -\frac{1}{2} & 0 \\ 0 & 0 & -\frac{1}{2} & \frac{3}{2} & -\frac{1}{2} \\ 0 & 0 & 0 & -\frac{1}{2} & 3 \end{pmatrix}.$$

E04: build a DIFFERENT toy stencil

AGH

Constant potential: not an oscillator

Take four sites, $h = 1$ and $V(x) = 0.75$ at every site. Thus $d_i = 1.75$ and $e_i = -0.5$.

$$\mathbf{v} = (1, 2, 3, 4)^T \quad \Longrightarrow \quad T\mathbf{v} = (0.75, 1.5, 2.25, 5.5)^T.$$

```
for (double &a:d) a=1/(h*h)+constant_potential;
for (double &a:e) a=-1/(2*h*h);
```

```
make bin/04_constant_potential_stencil
./bin/04_constant_potential_stencil
```

E04 checks the multiplication component by component. Your project must independently build $V(x) = x^2/2$.

www.agh.edu.pl

Why use a tridiagonal eigensolver?

Dense symmetric matrix (lecture background)

A Householder similarity transformation can reduce a general dense symmetric matrix to tridiagonal form while preserving eigenvalues. A specialised routine then obtains the spectrum.

$$A \xrightarrow[\text{Householder}]{Q^T A Q} T \xrightarrow[\text{specialised solver}]{} \Lambda.$$

Our advantage today

The second-difference Hamiltonian is **already tridiagonal**. We supply its n diagonal and $n-1$ subdiagonal entries directly to `LAPACKE_dstev`. Do not implement a general Householder/QR eigensolver for this lab.

Flat storage: one matrix, two arrangements

$$Z = \begin{pmatrix} 0 & 1 & 2 \\ 10 & 11 & 12 \\ 20 & 21 & 22 \end{pmatrix}.$$

Row-major: rows contiguous

`0,1,2, 10,11,12, 20,21,22`
Index: `z[i*ldz+k]`.
For $i = 1, k = 2$, offset $= 1 \cdot 3 + 2 = 5$.

Column-major: columns contiguous

`0,10,20, 1,11,21, 2,12,22`
Index: `z[i+k*ldz]`.
For $i = 1, k = 2$, offset $= 1 + 2 \cdot 3 = 7$.

If `ldz` $= 4$ the unused fourth slot in each stored column or row is padding, not another node.

E05: make memory indexing visible

```
constexpr int n=3, ldz=4;
row.fill(-99); col.fill(-99);
for (int i=0;i<n;++i)
  for (int k=0;k<n;++k) {
    row[i*ldz+k]=10*i+k;
    col[i+k*ldz]=10*i+k;
  }
```

Predict addresses for the SAME logical $Z_{1,2} = 12$

Padded row-major offset: $1 \cdot 4 + 2 = 6$.
Padded column-major offset: $1 + 2 \cdot 4 = 9$.

```
make bin/05_matrix_layout_indices
./bin/05_matrix_layout_indices
```

The full example checks every entry and confirms padding sentinels remain -99.

BLAS, LAPACK and LAPACKE: what you need

BLAS	Basic numerical matrix/vector kernels
LAPACK	Algorithms and routines for numerical linear algebra
LAPACKE	C interface used from C++ in this course

Build vs. run

`#include <lapacke.h>` declares the interface. A working g++ command still needs a compatible link library and its runtime dependencies.

Portable command route

`make lapacke` (builds the separately executable E06). Default flags: `-llapacke -llapack -lblas -lm`.

Provider-specific include paths, library locations and runtime paths may need overrides. See the README. No AGH classroom installation is assumed.

The complete LAPACKE dstev interface



```
lapack_int LAPACKE_dstev(int matrix_layout, char jobz,
                         lapack_int n, double* d, double* e,
                         double* z, lapack_int ldz);
```

`matrix_layout`	Row-major or column-major output storage
`jobz`	`'V'` vectors and values, `'N'` values only
`n`	Matrix dimension
`d`, `e`	Input diagonal/subdiagonal (overwritten)
`z`, `ldz`	Eigenvector columns and their storage stride
return `info`	0 success; negative bad argument; positive non-convergence

Use `lapack_int` consistently with the linked provider's integer ABI.

Which solver arrays change? Preserve originals

$$T = \begin{pmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{pmatrix}, \quad \underbrace{d = [2, 2, 2]}_{\text{3 diagonal values}}, \quad \underbrace{e = [-1, -1]}_{\text{2 physical offdiagonals}} .$$

Before the call

Keep untouched `d_original`, `e_original`.
Allocate output z with n^2 entries when requesting vectors.

After success

`d` holds ascending eigenvalues.
`e` is overwritten.
Logical columns of z are eigenvectors.

Allocating n elements for `e` is a safe teaching convention for some implementations; its last slot is not a new physical coupling.

Extract eigenvector columns, not rows

$$\underbrace{z_{i,k}}_{\text{component } i \text{ of state } k} = \begin{cases} \texttt{z[i+k*ldz]}, & \texttt{LAPACK_COL_MAJOR}, \\ \texttt{z[i*ldz+k]}, & \texttt{LAPACK_ROW_MAJOR}. \end{cases}$$

Logical (i,k)	$(0,0)$	$(1,2)$	$(2,1)$
COL with `ldz=3`	0	$1+2\cdot 3=7$	$2+1\cdot 3=5$
ROW with `ldz=3`	0	$1\cdot 3+2=5$	$2\cdot 3+1=7$

The physical interpretation never changes

The second index is the **eigenvector column**, not the spatial grid point. For a full $n \times n$ output without padding choose `ldz=n`.

Check status and reconstruct the ORIGINAL equation

```
auto d=d_original, e=e_original;
const lapack_int info=LAPACKE_dstev(
    LAPACK_COL_MAJOR,'V',n,d.data(),e.data(),
    z.data(),ldz);
if (info != 0) return 1; // Do not trust partial output.
```

Then check $T\mathbf{v}_k - \lambda_k \mathbf{v}_k$

Reconstruct each product using `d_original` and `e_original`. The overwritten `d` is now eigenvalues and must *not* be used as the original diagonal.

`info`=0 only reports algorithm completion, not physical accuracy or correct indexing.

E06: real LAPACKE eigensolver on a tiny toy

$$T = \begin{pmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{pmatrix}; \quad \lambda = 2 - \sqrt{2},\ 2,\ 2 + \sqrt{2}.$$

```
const lapack_int info=LAPACKE_dstev(
    LAPACK_COL_MAJOR,'V',n,d.data(),e.data(),
    z.data(),ldz);
if (info!=0) return 1;
```

```
make lapacke
./bin/06_lapacke_tridiagonal_eigensolver
```

Independent fixture

Expected eigenvalues approximately 0.5857864, 2, 3.4142136. The complete E06 file uses real LAPACKE, not a homemade eigensolver.

E06: verify the eigenpairs numerically

Three independent checks

1. Compare computed eigenvalues against $2 \pm \sqrt{2}$ and 2.
2. Compute $\max_{i,k} |(T\mathbf{v}_k)_i - \lambda_k v_{ik}|$ using *original* d, e.
3. Form all overlaps $\sum_i v_{ik} v_{il}$ and compare with δ_{kl}.

$$\| T\mathbf{v}_k - \lambda_k \mathbf{v}_k \|_\infty \simeq 0, \qquad \| Z^T Z - I \|_{\max} \simeq 0.$$

What makes this a meaningful test?

The eigenvalues come from an analytic formula, while the residual is formed from stored original matrix coefficients. A library success code alone would not check either result.

A wavefunction represents probability density

$$\int_{-\infty}^{\infty} |\psi(x)|^2 \, dx = 1 \quad \text{(continuous physical normalization).}$$

What a spatial grid changes

We hold samples $\psi_i \approx \psi(x_i)$ at equal spacing Δ. The integral becomes a weighted sum, with a length element for each sample:

$$\sum_{i=0}^{n-1} |\psi_i|^2 \, \Delta \approx 1 .$$

Why Δ matters

A sum of squared sample values is not yet an integral. Include the grid spacing as the spatial measure.

LAPACK unit length is not yet grid normalization



$$\text{LAPACK:} \quad \sum_i v_i^2 = 1,$$

$$\text{physical:} \quad \sum_i \psi_i^2 \Delta = 1, \qquad \boxed{\psi_i = \frac{v_i}{\sqrt{\Delta}}}.$$

Calculate with $\Delta = 1/4$ and $\mathbf{v} = (0.6, 0.8)^T$

$\psi = (1.2, 1.6)^T$, hence $\Delta(1.2^2 + 1.6^2) = \frac{1}{4}(1.44 + 2.56) = 1$.

Also normalise sampled *analytical* shapes using the same discrete measure before comparing curves.

E07: weighted norm, overlap and global sign

```
for (int i=0;i<2;++i)
    psi[i]=v[i]/std::sqrt(delta);
for (int i=0;i<2;++i)
    norm+=delta*psi[i]*psi[i];
const int flip=(opposite_overlap<0)?-1:1;
```

Run the complete vector-only example

```
make bin/07_grid_norm_overlap_sign
./bin/07_grid_norm_overlap_sign
```

Predict the checks

Weighted norm 1, orthogonal overlap 0, opposite-sign overlap -1, and overlap $+1$ after a *single* global sign reversal.

Exact oscillator energies and the report state labels

$$E_k = k + \tfrac{1}{2}, \qquad k = 0, 1, 2, \ldots \quad \text{(zero-based numerical index)}.$$

$$\boxed{\text{displayed state } s = k + 1, \qquad E_s = s - \tfrac{1}{2}}.$$

Displayed s	1	2	3	15	20	30
C++ $k = s - 1$	0	1	2	14	19	29
Exact E_s	0.5	1.5	2.5	14.5	19.5	29.5

Off-by-one trap

Displayed state 1 is the **ground state** $k = 0$, which uses Hermite polynomial H_0. State 30 uses eigenvector column 29 and H_{29}.

Physicists' Hermite polynomials

$$H_{-1}(x) = 0, \quad H_0(x) = 1, \qquad \boxed{H_k(x) = 2xH_{k-1}(x) - 2(k-1)H_{k-2}(x)}.$$

$$H_1(x) = 2x,$$
$$H_2(x) = 4x^2 - 2,$$
$$H_3(x) = 8x^3 - 12x.$$

Watch the convention

These are *physicists'* Hermite polynomials. Their recurrence and scale differ from the probabilists' convention. Iteration uses only the previous two values.

Hermite recurrence: every arithmetic step at $x = 0.5$

$$H_0(0.5) = 1,$$
$$H_1(0.5) = 2(0.5)H_0 - 0H_{-1} = 1,$$
$$H_2(0.5) = 2(0.5)H_1 - 2(1)H_0 = 1 - 2 = -1,$$
$$H_3(0.5) = 2(0.5)H_2 - 2(2)H_1 = -1 - 4 = -5.$$

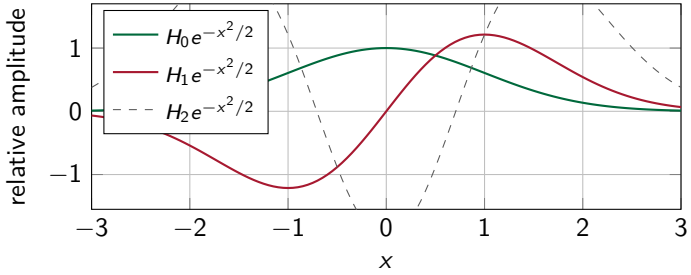
Independent polynomial check

$H_2(0.5) = 4(0.5)^2 - 2 = -1$; $H_3(0.5) = 8(0.5)^3 - 12(0.5) = 1 - 6 = -5$.

Predict `H0=1, H1=1, H2=-1, H3=-5` before trying E08.

Analytical wavefunction shapes and parity

$$\psi_k(x) \propto H_k(x) e^{-x^2/2}.$$



These are exact *unnormalized reference shapes*, not output from the oscillator matrix. Even k is even in x; odd k is odd.

E08: evaluate Hermite and Gaussian shapes

```
double hm2=0, hm1=1;
for (int j=1;j<=k;++j) {
    const double current=2*x*hm1-2*(j-1)*hm2;
    hm2=hm1; hm1=current;
}
return hm1;
```

Compile, then check at $x = 0.5$

```
make bin/08_hermite_gaussian_reference
./bin/08_hermite_gaussian_reference
```

Correct interpretation

Hermite values $1, 1, -1, -5$; multiplying by $e^{-x^2/2}$ creates *shape samples*, not automatically normalized wavefunctions.

Compare corresponding states with one global sign

$$\mathcal{O}_k = \Delta \sum_i \psi_{ik}^{\mathrm{num}} \psi_{ik}^{\mathrm{ref}}, \qquad \psi_{ik}^{\mathrm{num}} \leftarrow \begin{cases} -\psi_{ik}^{\mathrm{num}}, & \mathcal{O}_k < 0, \\ \psi_{ik}^{\mathrm{num}}, & \mathcal{O}_k \geq 0. \end{cases}$$

Do the comparisons in this order

(1) Pair matching k and $s = k + 1$; (2) discretely normalize both sampled vectors; (3) measure their overlap; (4) optionally reverse *one whole vector*.

Do not flip node by node

Changing signs at individual nodes destroys a wavefunction. A nearly zero overlap may signal wrong state pairing or a poorly resolved vector; investigate before deciding a sign.

A trustworthy eigenpair needs a residual

$$r_i = (T\mathbf{v}_k)_i - \lambda_k v_{ik}, \qquad \|r\|_\infty = \max_i |r_i|.$$

$$\eta_k = \frac{\|r\|_\infty}{(\|T\|_\infty + |\lambda_k|)\|\mathbf{v}_k\|_\infty} \quad \text{if denominator} > 0.$$

Also check

- $\mathbf{v}_k^T \mathbf{v}_l \approx \delta_{kl}$ (solver output).
- $\Delta \sum_i (\psi_i^{(k)})^2 \approx 1$ (physical norm).
- Ascending eigenvalues and finite values.

In the zero-scale special case, report a raw residual separately rather than dividing by zero.

Diagnose the FIRST incorrect quantity

Symptom	First item to inspect
Wrong matrix coefficients	n, Δ, x_i, sign of subdiagonal
Wrong eigenvector shape	Column index k, `1dz`, storage layout
Huge residual	Original d, e versus overwritten solver arrays
Wrong ground-state curve	$s = 1 \Leftrightarrow k = 0 \Leftrightarrow H_0$
Norm depends strongly on n	Missing Δ weighting
Wrong curve sign	Global overlap; not node-wise changes
Higher states disagree	Grid resolution and/or finite box

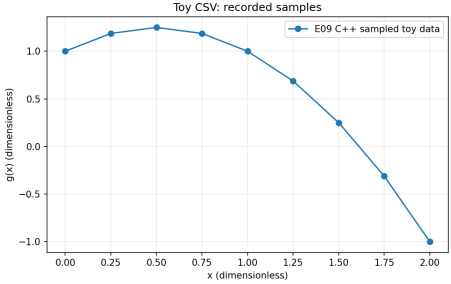
Remember the two physical errors

Finite differences cause $O(\Delta^2)$ local stencil error. Truncating the infinite domain adds box effects that are not cured merely by increasing n at fixed $x_{\max}$.

E09: real CSV records become a plot

Fixed unrelated toy
$g(x) = 1 + x - x^2$

The standalone C++ writer saves 9 points, $x = 0, 0.25, \ldots, 2$; columns `x,y`. The plotter reads *these recorded values*; it does not regenerate the function.




```
make bin/09_toy_csv_output
./bin/09_toy_csv_output data/toy_run02.csv
python3 plotting/plot_toy_csv.py \
   data/toy_run02.csv \
   --output plots/toy_run02.png
```

Existing files are protected from accidental replacement (plot override: `-overwrite`). The embedded plot was generated from actual E09 C++ CSV.

How to design the required project figures

Energy spectrum figure

- Horizontal axis: displayed state s.
- Vertical axis: energy E_s.
- Numerical points and exact reference clearly distinguished.
- Keep n and $x_{\max}$ visible.

Wavefunction figures

- x_i on the horizontal axis, $\psi_s(x_i)$ vertically.
- Compare the same state and matching grid normalization.
- State s, grid size n, sign alignment and legend.

Figure-design guidance, NOT precomputed homework results

Do not plot invented values. Use actual solver and analytical arrays saved by your own C++ project; label line joins as sampled data, not an exact continuous numerical curve.

Official Task 1 + Task 2: grid and Hamiltonian arrays

AGH

Required initial inputs (handout)

$n = 100$, $x_{\max} = 10$, $\Delta = 20/(100-1)$, $x_i = -10 + i\Delta$, $i = 0, \ldots, 99$.

Your independent implementation

1. Create the endpoint-inclusive grid.
2. Apply the derived Hamiltonian formulas to fill `d` and `e`.
3. Print/check first, middle and last entries before calling the solver.

See S19–S27 and E02–E04 for *different* teaching fixtures.

Official Task 2: numerical energies and states

Complete these steps in your own C++ program

1. Preserve original arrays for any residual checks.
2. Choose the row/column layout, allocate and document `z` and `ldz`.
3. Call `LAPACKE_dstev` and inspect the return code.
4. Save energies, extract *columns* in the chosen layout.
5. Normalize numerically sampled wavefunctions with Δ and save the data.

Verification checkpoint

Before plotting, check eigenvalue ordering, $H\mathbf{v}_k - E_k\mathbf{v}_k$, weighted norms and finite outputs.

E05 and E06 explain storage and the genuine solver on unrelated small matrices; E07 explains normalization.

Official Tasks 3–4: analytical reference and energies

In the student oscillator program

1. At each official grid point calculate $H_k(x_i)e^{-x_i^2/2}$ with the recurrence.
2. Discretely normalize the sampled analytical wavefunctions.
3. Save consistent exact/numerical data with your documented file convention.
4. Make an energy-spectrum figure with numerical and exact energies.

$$s = 1, 2, \ldots \iff k = s - 1, \qquad E_s = s - \frac{1}{2}.$$

Transfer the lesson, not the finished work

E08 is a small Hermite evaluator; E09 demonstrates CSV/plotting on an unrelated polynomial. The actual QHO data creation and comparison remain yours.

HOME REPORT: required wavefunction comparisons

AGH

$\boxed{s = 1,\ 2,\ 3,\ 15,\ 20,\ 30}$ (zero-based columns $k = s - 1$).

Repeat the full calculation at finer resolution

At home, use $n = 1000$ with the same $x_{\max} = 10$. Produce new energy-spectrum and wavefunction figures, then compare with the original $n = 100$ calculations.

Answer the handout's questions

Which eigenvalues and eigenvectors agree best with analytical states? Why do the highest states tend to differ? Which results improve after refining n and why? Discuss both second-difference error and the finite computational box.

Independent project: checkpoints and source map

Checkpoint	Evidence you can explain
1. Grid and matrix	x_i, d_i, e_i and physical index convention
2. Diagonalization	Solver status, column extraction, numerical norms
3. Analysis and plots	Hermite recurrence, file provenance, spectral checks
4. Home report	Both n choices; six states; explained errors

Reliable working order

Predict → implement one stage → test → save → plot → interpret. Ask for guidance at the first value you cannot explain.

Sources: [H] Chwiej, *Diagonalization of symmetric tridiagonal matrix*, 19 March 2025, pp. 1–5; [L] Chwiej, *Diagonalization of dense matrices*, pp. 1–30; [N] Netlib LAPACKE documentation; [T] accompanying example sources.

OPTIONAL E10: convergence in a DIFFERENT box

Free particle; interior-node Dirichlet box (NOT the oscillator lab)

For a box $(0, L)$ with zero boundary values, take N interior sites, $h = L/(N+1)$, $V = 0$ and tridiagonal $d_i = 1/h^2$, $e_i = -1/(2h^2)$.

$$E_1^{\text{continuum}} = \frac{\pi^2}{2L^2}, \qquad E_1^{\text{discrete}} = \frac{1 - \cos\big(\pi/(N+1)\big)}{h^2}.$$

E10 verifies second-order convergence

With $L = \pi$ and $N = 7, 15, 31$, the independently known continuum ground energy is $1/2$; the discrete error falls by roughly 4 when the spacing halves.


THANK
YOU!