QR factorization example using Householder reflections.

$H_v = I - 2vv^T$ where

$$v = \frac{a_1 - ce_1}{\|a_1 - ce_1\|} \qquad c = \pm \|a_1\| .$$

Choose + or - minus to maximize $\|a_1 - ce_1\|$

Example:

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 8 & 9 & 4 \\ 7 & 6 & 5 \end{bmatrix}$$

$$A = \left[a_1 \middle| a_2 \middle| a_3 \right]$$

```
julia> A=[1 2 3; 8 9 4; 7 6 5]
3×3 Matrix{Int64}:
 1  2  3
 8  9  4
 7  6  5
```

```
julia> A[:,1]
3-element Vector{Int64}:
 1
 8
 7

julia> e1=[1,0,0]
3-element Vector{Int64}:
 1
 0
 0

julia> using LinearAlgebra
```

```
julia> c=norm(A[:,1])
10.677078252031311

julia> v=A[:,1]-c*e1
3-element Vector{Float64}:
 -9.677078252031311
  8.0
  7.0

julia> norm(v)
14.375181511756205

julia> v=A[:,1]+c*e1
3-element Vector{Float64}:
 11.677078252031311
  8.0
  7.0

julia> norm(v)
15.790951728887737
```

which is bigger?

bigger

make it into a unit vector...

```
julia> v=v/norm(v)
3-element Vector{Float64}:
 0.7394790670323838
 0.5066192422946184
 0.4432918370077911

julia> H1=I-2*v*v'
3×3 Matrix{Float64}:
 -0.0936586  -0.749269  -0.65561
 -0.749269    0.486674  -0.44916
 -0.65561    -0.44916    0.606985
```

supposed to have that

$$H_1 A = \left[\begin{array}{c|c|c} c & H_1 a_2 & H_1 a_3 \\ 0 & & \\ 0 & & \end{array}\right]$$

```
julia> H1*A
3×3 Matrix{Float64}:
 -10.6771       -10.8644     -6.5561
  -4.44089e-16    0.186566   -2.54691
  -8.88178e-16   -1.71176    -0.728548
```

Now do the same thing to transform this submatrix so it looks like $\begin{bmatrix} c & ? \\ 0 & ? \end{bmatrix}$

these are equal 0 up to rounding...

```
julia> H1*A
3×3 Matrix{Float64}:
 -10.6771       -10.8644     -6.5561
  -4.44089e-16    0.186566   -2.54691
  -8.88178e-16   -1.71176    -0.728548

julia> A2=(H1*A)[2:3,2:3]
2×2 Matrix{Float64}:
  0.186566  -2.54691
 -1.71176   -0.728548
```

first column

$$v = \frac{a_1 - ce_1}{\|a_1 - ce_1\|} \qquad c = \pm \|a_1\|$$

```
julia> e1=[1,0]
2-element Vector{Int64}:
 1
 0

julia> c=norm(A2[:,1])
1.7218920641845576

julia> norm(A2[:,1]-c*e1)
2.2994201346160796

julia> norm(A2[:,1]+c*e1)
2.5636528952510704
```

← e_1 in 2D

← bigger

```
julia> v2=A2[:,1]+c*e1
2-element Vector{Float64}:
  1.9084576507534088
 -1.7117551117522556

julia> v2=v2/norm(v2)
2-element Vector{Float64}:
  0.7444290349480033
 -0.667701588979976
```

Now need to find H_2, but I want it to act on the 3x3 matrix...

```
julia> v2pad=[0; v2]
3-element Vector{Float64}:
  0.0
  0.7444290349480033
 -0.667701588979976

julia> H2=I-2*v2pad*v2pad'
3×3 Matrix{Float64}:
  1.0  -0.0       0.0
 -0.0  -0.108349  0.994113
  0.0   0.994113  0.108349
```

Supposed to be that

$$H_2 H_1 A = R$$

↑ upper triangular...

```
julia> H2*H1*A ≈ R
3×3 Matrix{Float64}:
 -10.6771       -10.8644       -6.5561
  -8.88178e-16   -1.72189      -0.448303
  -8.88178e-16    4.44089e-16  -2.61086
```

↑ this part is 0 up to rounding errors.

```
julia> R=H2*H1*A
3×3 Matrix{Float64}:
 -10.6771       -10.8644       -6.5561
  -8.88178e-16   -1.72189      -0.448303
  -8.88178e-16    4.44089e-16  -2.61086
```

So what's the QR factorization?

$$H_2 H_1 A = R$$

$$A = H_1^{-1} H_2^{-1} R$$ ← Since H's are orthogonal.

$$A = H_1^T H_2^T R$$ ← Since the H's are symmetric.

$$A = H_1 H_2 R$$

$$Q = H_1 H_2$$

```
julia> Q=H1*H2
3×3 Matrix{Float64}:
 -0.0936586  -0.570568  -0.815892
 -0.749269   -0.499247   0.435143
 -0.65561     0.652077  -0.38075

julia> Q*R
3×3 Matrix{Float64}:
 1.0  2.0  3.0
 8.0  9.0  4.0
 7.0  6.0  5.0
```

This is the QR factorization of A

```
julia> A
3×3 Matrix{Int64}:
 1  2  3
 8  9  4
 7  6  5

julia> qr(A)
LinearAlgebra.QRCompactWY{Float64, Matrix{Float64}}
Q factor:
3×3 LinearAlgebra.QRCompactWYQ{Float64, Matrix{Float64}}:
 -0.0936586  -0.570568  -0.815892
 -0.749269   -0.499247   0.435143
 -0.65561     0.652077  -0.38075
R factor:
3×3 Matrix{Float64}:
 -10.6771  -10.8644   -6.5561
   0.0      -1.72189  -0.448303
   0.0       0.0      -2.61086
```

Julia has QR built in...

```
julia> R=H2*H1*A
3×3 Matrix{Float64}:
 -10.6771        -10.8644        -6.5561
  -8.88178e-16    -1.72189       -0.448303
  -8.88178e-16     4.44089e-16   -2.61086
```

Linear Algebra:

Gaussian elimination: $A = P L U$

upper triangular matrix

lower triangular with 1's on the diagonal

permutation matrix that encodes the row swaps during the elimination

Gram Schmidt: $A = \tilde{Q}\tilde{R}$ ← upper triangular.

A: $m \times n$, $\tilde{Q}$: $m \times n$, $\tilde{R}$: $n \times n$

$\tilde{Q}$: matrix with orthonormal columns.. $\tilde{Q}^T\tilde{Q} = I$

Householder reflectors: $A = QR$ ← upper triangular with a block of 0's at the bottom

A: $m \times n$, Q: $m \times m$, R: $m \times n$

Q: orthogonal matrix. Square and $Q^TQ = I$

Eigenvalue - Eigenvectors: $A = SDS^{-1}$

S: matrix of eigenvectors. D: diagonal matrix of eigenvalues

Spectral theorem:

$A = A^T$

orthonormal basis of eigenvectors

$A = QDQ^{-1}$

Q: matrix of eigenvectors. D: diagonal matrix of eigenvalues..

But eigenvalues are real and Q is an orthogonal matrix. $Q^TQ = I$.

Singular value decomposition

$B = A^TA$

and note that $B^T = B$

$A = U\Sigma V^T$ ← orthogonal matrix

U: orthogonal matrix. Σ: diagonal matrix of singular values ($\sqrt{\text{eigenvalues of } B}$)

Numerical Quadrature. Approximating $\int_a^b f(x)dx$.

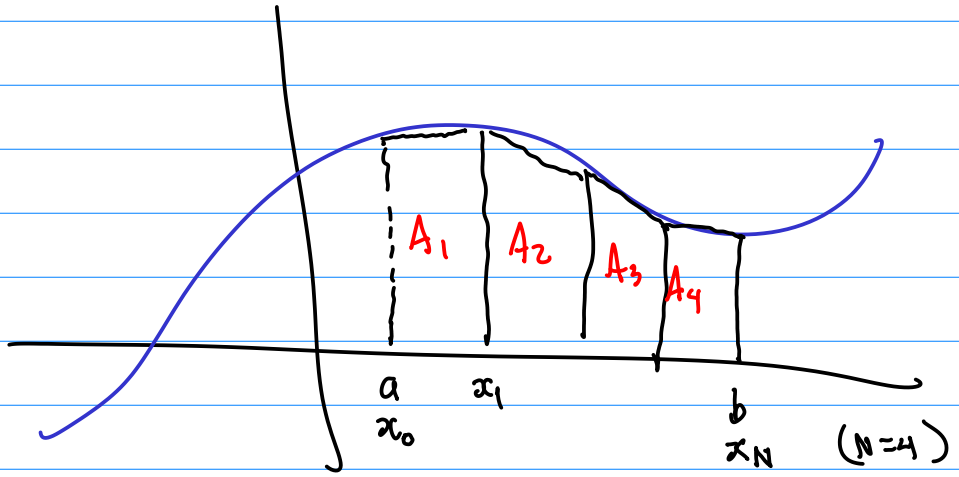
STEP 30

NUMERICAL INTEGRATION 1
The trapezoidal rule

Use trapezoids to approximate areas...

$$A = h\frac{f(x_i)+f(x_{i+1})}{2}$$

To approximate area under a curve use lots of trapezoids...



$$x_i = x_0 + hi \quad \text{where} \quad h = \frac{b-a}{N}$$

$$\int_a^b f(x)\,dx \approx h\frac{f(x_0)+f(x_1)}{2} + h\frac{f(x_1)+f(x_2)}{2} + \cdots + h\frac{f(x_{N-1})+f(x_N)}{2}$$

$$\int_a^b f(x)\,dx \approx \sum_{j=0}^{N-1} h\,\frac{f(x_j)+f(x_{j+1})}{2}$$

we know that $\lim_{N\to\infty} \sum_{j=0}^{N-1} h\,\frac{f(x_j)+f(x_{j+1})}{2} = \int_a^b f(x)\,dx \ldots$

What is the error for finite N?

Total error

$$E \approx \int_a^b f(x)\,dx - \sum_{j=0}^{N-1} h\,\frac{f(x_j)+f(x_{j+1})}{2}$$

$$= \sum_{j=0}^{N-1} \left[\int_{x_j}^{x_{j+1}} f(x)\,dx - h\,\frac{f(x_j)+f(x_{j+1})}{2} \right] = \sum_{j=0}^{N-1} E_j$$

The truncation error

$$E_j = \int_{x_j}^{x_{j+1}} f(x)\,dx - h\,\frac{f(x_j)+f(x_{j+1})}{2}$$