If $\|a_1 - \|a_1\| e_1\| > \|a_1 + \|a_1\| e_1\|$

then $c = \|a_1\|$

otherwise $c = -\|a_1\|$

choose $\pm$ to make norm as large as possible

$$\left\| \begin{bmatrix} a_{11} \\ a_{21} \\ a_{31} \\ \vdots \\ a_{m1} \end{bmatrix} - \begin{bmatrix} \pm\|a_1\| \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \right\| = \sqrt{(a_{11} \pm \|a_1\|)^2 + a_{21}^2 + a_{31}^2 + \cdots + a_{m1}^2}$$

don't depend on the sign of c

I want $|a_{11} \pm \|a_1\||$ as large as possible.

If $a_{11} \geq 0$ then $|a_{11} + \|a_1\||$ is bigger

$a_{11} < 0$ then $|a_{11} - \|a_1\||$ is larger.

If $a_{11} \in \mathbb{C}$ then what? $a_{11} = \alpha_{11} + i\beta_{11}$ for $\alpha_{11}, \beta_{11} \in \mathbb{R}$

$$|a_{11} \pm \|a_1\|| = \sqrt{(\alpha_{11} \pm \|a_1\|)^2 + \beta_{11}^2}$$

complex real

If $\operatorname{Re} a_{11} \geq 0$ then $|a_{11} + \|a_1\||$ is bigger

$\operatorname{Re} a_{11} < 0$ then $|a_{11} - \|a_1\||$ is larger.

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$ then $w = a_1 + \|a_1\| e_1$

```
julia> e2=[1,0]
2-element Vector{Int64}:
 1
 0

julia> A2=(H*A)[2:3,2:3]
2×2 Matrix{Float64}:
 -0.0859656  -0.171931
 -0.90044    -1.80088
```

$$I - 2\begin{bmatrix}0\\ v_2\end{bmatrix}\begin{bmatrix}0\\ v_2\end{bmatrix}^T$$

```
julia> v2big=[0; v2]
3-element Vector{Float64}:
  0.0
 -0.7399454417565073
 -0.6726668887523506

julia> H2big=I-2*v2big*v2big'
3×3 Matrix{Float64}:
 1.0   0.0         0.0
 0.0  -0.0950385  -0.995474
 0.0  -0.995474    0.0950385
```

```
julia> w2=A2[:,1]-norm(A2[:,1])*e2
2-element Vector{Float64}:
 -0.9904996037569178
 -0.9004397475413497

julia> v2=w2/norm(w2)
2-element Vector{Float64}:
 -0.7399454417565073
 -0.6726668887523506

julia> H2=I-2*v2*v2'
2×2 Matrix{Float64}:
 -0.0950385  -0.995474
 -0.995474    0.0950385

julia> H2*A2
2×2 Matrix{Float64}:
 0.904534      1.80907
 2.77556e-17   1.72085e-15
```

```
julia> H2big*H*A
3×3 Matrix{Float64}:
 -8.12404       -9.60114       -11.0782
 -8.88178e-16    0.904534        1.80907
 -8.88178e-16   -4.44089e-16     8.88178e-16
```

same ...

do this for every column of $A \in \mathbb{R}^{m\times n}$ or only n column if $m<n$.

$$H_\eta \cdots H_3 H_2 H_1 A = R = \begin{cases} \begin{bmatrix} * & * & * & * \\ 0 & * & * & * \\ 0 & 0 & * & * \\ 0 & 0 & 0 & * \end{bmatrix} & m = n \\ \begin{bmatrix} * & * & * & * \\ 0 & * & * & * \\ 0 & 0 & * & * \\ 0 & 0 & 0 & * \\ 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 \end{bmatrix} & m > n \\ \begin{bmatrix} * & * & * & * & * & \cdots & * \\ 0 & * & * & * & * & \cdots & * \\ 0 & 0 & * & * & * & \cdots & * \\ 0 & 0 & 0 & * & * & \cdots & * \end{bmatrix} & m < n \end{cases}$$

works fine..

overdetermined system where we solve least squares

underdetermined so many solutions in this case pick solution of least norm... later...

From the text they found that $c = \pm \|a_1\|$ much quicker using the fact that orthogonal matrices preserve length.

$Ha_1 = ce_1$

$\|a_1\| = \|Ha_1\| = \|ce_1\|$

$\|a_1\| = |c|\,\|e_1\| = |c|$

Thus $|c| = \|a_1\|$

or $c = \pm \|a_1\|$

Quicker way to deduce this than what we did last week.

```
"""
    qrfact(A)

QR factorization by Householder reflections. Returns Q and R.
"""
function qrfact(A)
    m,n = size(A)
    Qt = diagm(ones(m))
    R = float(copy(A))
    for k in 1:n
        z = R[k:m,k]
        w = [ -sign(z[1])*norm(z) - z[1]; -z[2:end] ]
        nrmw = norm(w)
        if nrmw < eps() continue; end    # skip this iteration
        v = w / nrmw;
        # Apply the reflection to each relevant column of A and
        for j in 1:n
            R[k:m,j] -= v*( 2*(v'*R[k:m,j]) )
        end
        for j in 1:m
            Qt[k:m,j] -= v*( 2*(v'*Qt[k:m,j]) )
        end
    end
    return Qt',triu(R)
end
```

bug

k^{th} column

choose $\pm$ for c except problem if $z[1] = 0$ (need to fix this)

In case columns weren't linearly indep.

R_k by H_k

H_{k-1} by H_k

zero out things supposed to be zero...

$H = I - 2vv^T$

$H^T = (I - 2vv^T)^T$

$= I^T - (2vv^T)^T$

$= I - 2vv^T v^T$

$= I - 2vv^T = H$

Note that

$$\underbrace{H_n \cdots H_3 H_2 H_1}_{Qt} A = R$$

orthogonal

$$H_{n-1} \cdots H_3 H_2 H_1 A = H_n^{-1} R = H_n^T R = H_n R$$

$$A = \underbrace{H_1 H_2 H_2 \cdots H_n}_{Q} R$$

$$\left(H_n \cdots H_3 H_2 H_1\right)^T = H_1^T H_2^T \cdots H_n^T = H_1 H_2 \cdots H_n$$

From the book.

```
julia> Q,R=qrfact(A)
([-0.12309149097933281 0.9045340337332
9659639173309 0.3015113445777639 0.8164
 -0.30151134457776396 -0.4082482904638
36296387953 -11.078234188139945; 0.0 0.
5798; 0.0 0.0 -6.661338147750939e-16])

julia> Q
3×3 adjoint(::Matrix{Float64}) with elt
 -0.123091   0.904534  -0.408248
 -0.492366   0.301511   0.816497
 -0.86164   -0.301511  -0.408248

julia> R
3×3 Matrix{Float64}:
 -8.12404  -9.60114   -11.0782
  0.0       0.904534    1.80907
  0.0       0.0        -6.66134e-16
```

Q ok.

```
julia> H*H2big
3×3 Matrix{Float64}:
 -0.123091   0.904534   0.408248
 -0.492366   0.301511  -0.816497
 -0.86164   -0.301511   0.408248
```

same:

```
julia> H2big*H*A
3×3 Matrix{Float64}:
 -8.12404       -9.60114       -11.0782
 -8.88178e-16    0.904534        1.80907
 -8.88178e-16   -4.44089e-16     8.88178e-16
```

```
julia> A=[0 2 3; 4 5 6; 7 8 9]
3×3 Matrix{Int64}:
 0  2  3
 4  5  6
 7  8  9

julia> Q,R=qrfact(A)
([1.0 0.0 0.0; 0.0 0.529998940003180
040050881 -0.5299989400031797], [0.0
.811978376064873; 0.0 0.0 0.31799936

julia> Q
3×3 adjoint(::Matrix{Float64}) with
 1.0  0.0        0.0
 0.0  0.529999   0.847998
 0.0  0.847998  -0.529999

julia> R
3×3 Matrix{Float64}:
 0.0  2.0       3.0
 0.0  9.43398  10.812
 0.0  0.0       0.317999

julia> Q*R
3×3 Matrix{Float64}:
 0.0  2.0  3.0
 0.0  5.0  6.0
 0.0  8.0  9.0
```

$a_{11}=0$ so bug with sign in line 12 QR doesn't work

Extra Credit: Fix the program in the book so it works in this case

$A = QR$

$m\times n \quad m\times m \quad m\times n$

overdetermined $m > n$

If $m \gg n$ then Q is a huge matrix.

You don't want to make Q in practice if m is too big...

$H_k = I - 2w_k w_k^T$ instead store the v_k's

$v_k \in \mathbb{R}^m$ for $k = 1, \ldots, n$

Instead of Qx do $H_1(H_2 \cdots (H_n x))$ which is efficient

also instead of $Q^T b$ do $H_n(H_{n-1} \cdots (H_1 b))$ which is efficient