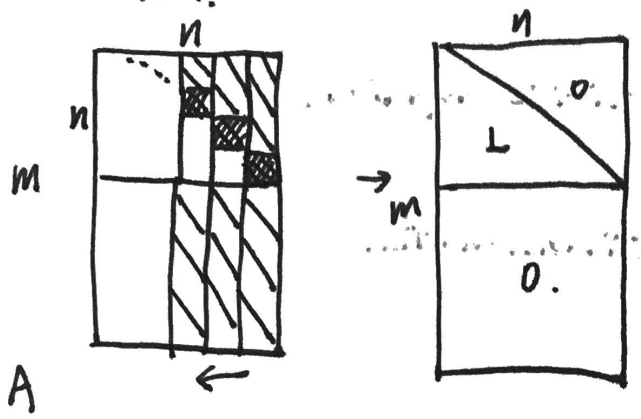
P 5.4.4.



Choose ▩ in the diagonal of $A(1:n, 1:n)$ as the first element in x, and

▨ $A(1:i-1, i)$ and $A(n+1:m, i)$ as the rest part of x. to make house holder matrix.

```
function [Q, L] = ql(A)
    m = size(A, 1);
    n = size(A, 2);
    Q = eye(m);

    for i = n:-1:1
        x = [A(i, i); A(1:i-1, i); A(n+1:m, i)];
        [rho, u] = make_house(x);

        Anew = apply_house(rho, u, [A(i,:); A(1:i-1, :); A(n+1:m, :)]);
        Qnew = apply_house(rho, u, [Q(i,:); Q(1:i-1, :); Q(n+1:m, :)]);

        A(i, :) = Anew(1, :);
        A(1:i-1, :) = Anew(2:i, :);
        A(n+1:m, :) = Anew(i+1:end, :);

        Q(i, :) = Qnew(1, :);
        Q(1:i-1, :) = Qnew(2:i, :);
        Q(n+1:m, :) = Qnew(i+1:end, :);

    end
    L = A;
    Q = Q';
end.
```

P8.2.1

$A_0 \in R^{n \times n}$, symmetric and positive definite.

$A_{k-1} = R^H R$ (Cholesky Faccorization)

then $R A_{k-1} R^{-1} = R R^H = A_k$ (similarity transformation)

$A_k = (R^{k-1} \cdots R^1) A_0 (R^{k-1} \cdots R^1)^{-1}$

$(R^{k-1} \cdots R^1)$ increases with iterations, but A_k converge to to the diagonal singular-values matrix.

```
function[e] = eigchol(A)
    n = size(A,1);
    x0 = diag(A);
    x1 = zeros(n,1);
    while norm(x1-x0) > 1e-13
        R = mychol(A);
        x0 = diag(A);
        A = R' * R;
        x1 = diag(A)
    end
    e = diag(A);
end
```

We can use single shift in the iteration to make the convergence faster since each iteration the diagonal values are approaching to the real eigenvalues, we can make a guess based on them.

P3: My MATLAB version is R2015b (8.6.0.267246) 64 bit mac maci64.

`m = size(A,2);` `nnz(A(:,col))` → 4.

I first experiement with row = 600, col = 188.

```
tic
r = A(1,col);
toc
```
→ $t_1 = 0.0024s$

```
tic
r = A(m,col);
toc
```
→ $t_2 = 2.4398e-5$

```
tic
r = A(row,col);
toc
```
→ $t_3 = 1.5846e-5$

When I change row and column, the time of t1, t2, t3 gives me almost the same trend as this. Thus, I think MATLAB does exploit special cases like row = 1 and row = m.

Then I modify A by changing col = 188 to a dense vector.

```
A(:,col) = rand(m,1);
nnz(A(:,col));
```
→ 916428.

And repeat the operation above.

$t_1 =$ 1.9194 e-04.

$t_2 =$ 1.0689 e-04.

$t_3 =$ 1.3098 e-04.

We can see that the time t2, t3 increase about 10 times. However, the non zero entries increase about.

$\frac{916428}{4} \simeq 23,0000$ times.

Thus, I think MATLAB might not use a linear search of each column and probably use binary search instead.