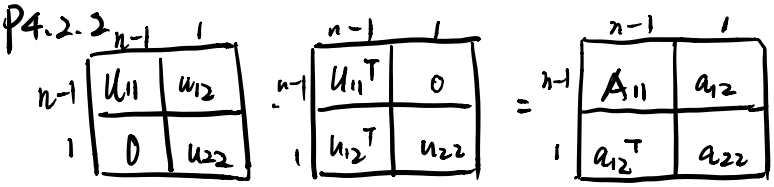
P4.2.2



① $U_{11} \cdot U_{11}^T + u_{12} \cdot u_{12}^T = A_{11}$

② $u_{12} \cdot u_{22} = a_{12}$

③ $u_{22} \cdot u_{12}^T = a_{12}^T$

④ $u_{22} \cdot u_{22} = a_{22}$

First compute $u_{22} = \sqrt{a_{22}}$, then $u_{12} = a_{12}/u_{22}$.

finally recursively compute $U_{11} U_{11}^T = A_{11} - u_{12} u_{12}^T$.

In Matlab:

```
function [R] = mychol(A)
n = size(A,1);
if n==1
    R(n,n) = sqrt(A(n,n));
end
if n>1
    R(n,n) = sqrt(A(n,n));
    R(1:n-1,n) = A(1:n-1,n)/R(n,n);
    R(1:n-1,1:n-1) = mychol( A(1:n-1,1:n-1) - R(1:n-1,n)*R(1:n-1,n)');
end
end.
```

P3.5.1

Table Without comparing with x=A\b.

	RA(sum)	RAC(max)	AC(sum)	AC(max)	RAC(sum)	RAC(max)	Rep(sum)	Rep(max)	D=A
rel.res	8	1	7	13	6	1	4	0	60
rel.err	15	12	8	1	15	10	12	15	12
cond	4	0	7	0	0	0	89	0	0

We can see from the above table, there are still 60 out of 100 problems where the relative residuals are better with no scaling, and 12 out of 100 problems where the relative errors are better with no scaling methods. It shows that simple row/column/row-column/repeated scaling do not "solve" the scaling problem.

With those problems where scaling methods do improve the result, I can't reach a general conclusion which one works the best. It depends on what kinds of problem you are solving. In some applications we need small residuals and others we need accuracy. All these factors affect which scaling methods we will choose. And another thing is that repeated scaling method with sum gives the best condition number.

With x=A\b

	RA(sum)	RAC(max)	AC(sum)	AC(max)	RAC(sum)	RAC(max)	Rep(sum)	Rep(max)	D=A	x=A\b
rel.res	1	1	6	9	3	1	1	0	54	24
rel.err	5	7	3	3	5	5	4	3	16	49
cond	4	0	7	0	0	0	89	0	0	0

This table includes x=A\b. We can see it doesn't beat the other scaling methods completely. Maybe because it's internal scaling is simple, by changing to other scaling methods, the result can still be improved.