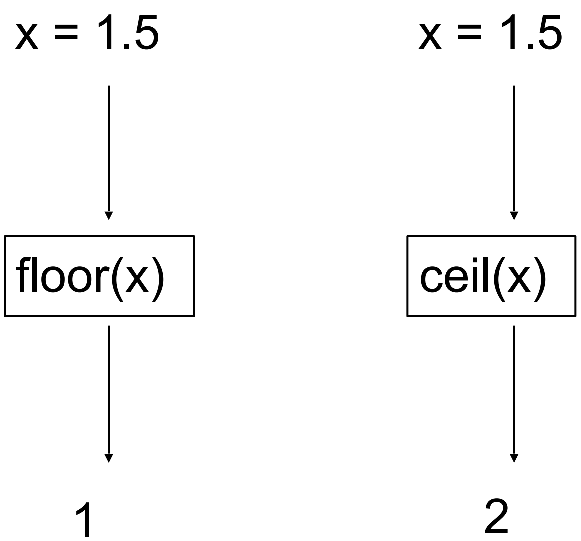
Lecture 5 FOR Looping

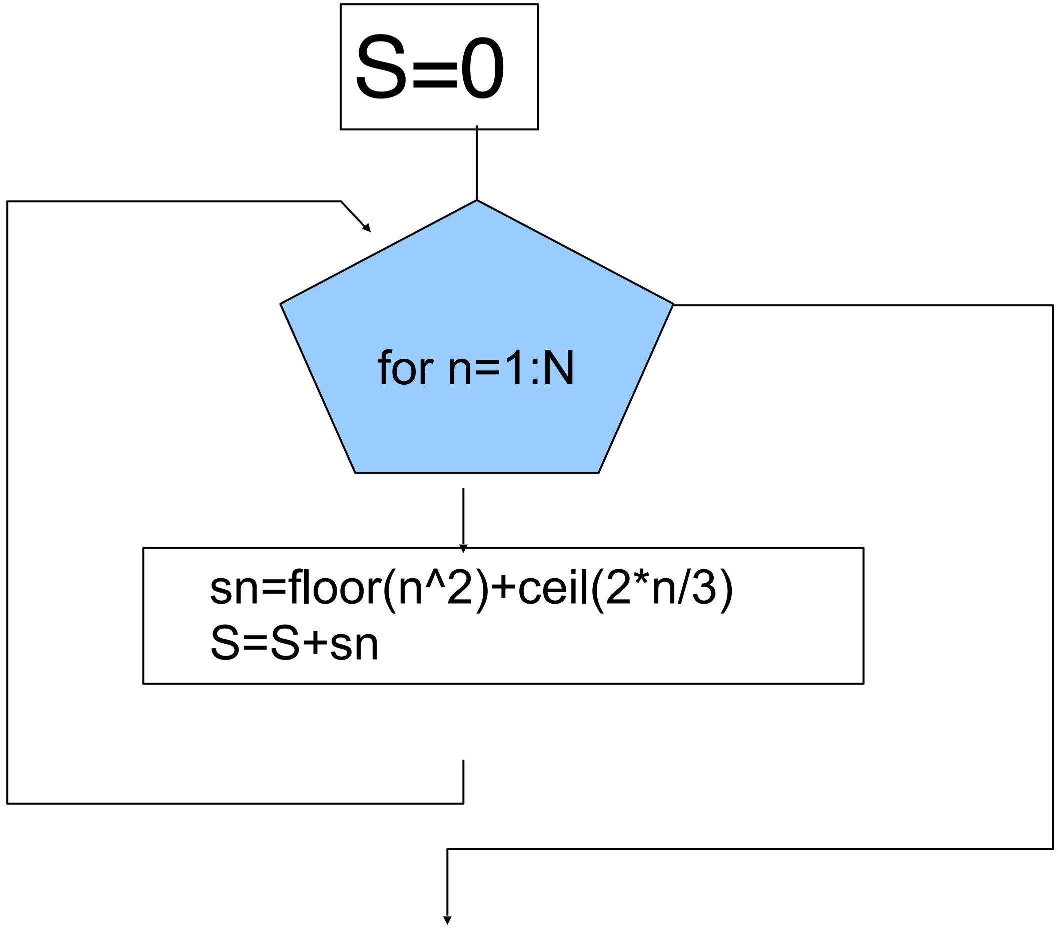
- Series sum of ceils and floors
- Symbolic differentiation
- Counting ATCG
- Markov Chain

Summation

$$S(N) = \sum_{n=1}^{N}\left(\left\lfloor \frac{n^2}{5} \right\rfloor + \left\lceil \frac{2*n}{3} \right\rceil\right)$$



Flow chart



Series

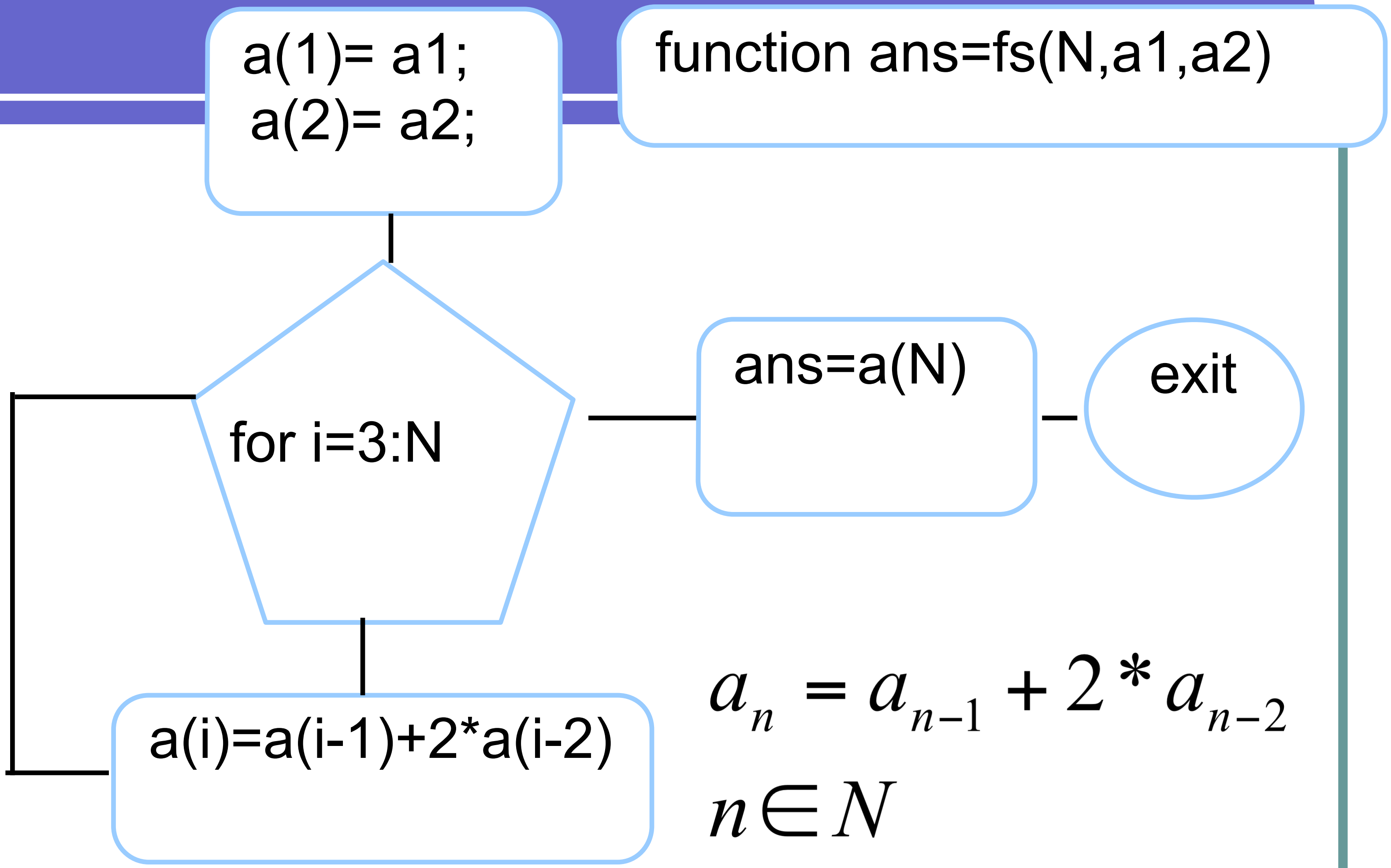
$$a_n = a_{n-1} + 2 * a_{n-2}$$

$$n \in N$$

Problem

- INPUT: a_1, a_2 and n
- OUTPUT: a_n

- Draw a flow chart to determine a_n by a for-loop
- Write a matlab function to calculate a_n for given a_1 , a_2 and n



$$a_n = a_{n-1} + 2 * a_{n-2}$$

$$n \in N$$

Symbolic Differentiation

```
>> x=sym('x');
>> diff(tanh(x))

ans =

1-tanh(x)^2
```

Symbolic differentiation

```
>> x=sym('x');
>> diff(x.^3+2*x)

ans =

3*x^2+2
```

Symbolic Differentiation

demo_diff.m

- Input a string, fstr
- Plot the function specified by fstr
- Plot the derivative of the given function

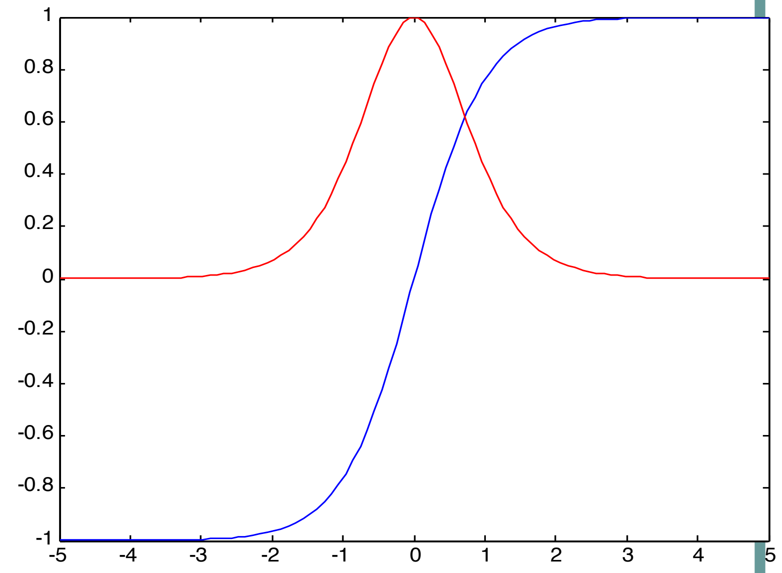
- MATLAB Web Server Demos

Example

function of x:tanh(x)

fx1 =

Inline function:
fx1(x) = 1-tanh(x).^2



Histogram

- Sample space, {1,2,3,4,5,6}
- Count occurrences of possible outcomes in a series

A series

```
n=10;
s=ceil(rand(1,10)*6);
```

Histogram

s =

6 2 4 3 6 5 3 1 5 3

↓

count=hist(s,[1 2 3 4 5 6])

↓

count =

1 1 3 1 2 2

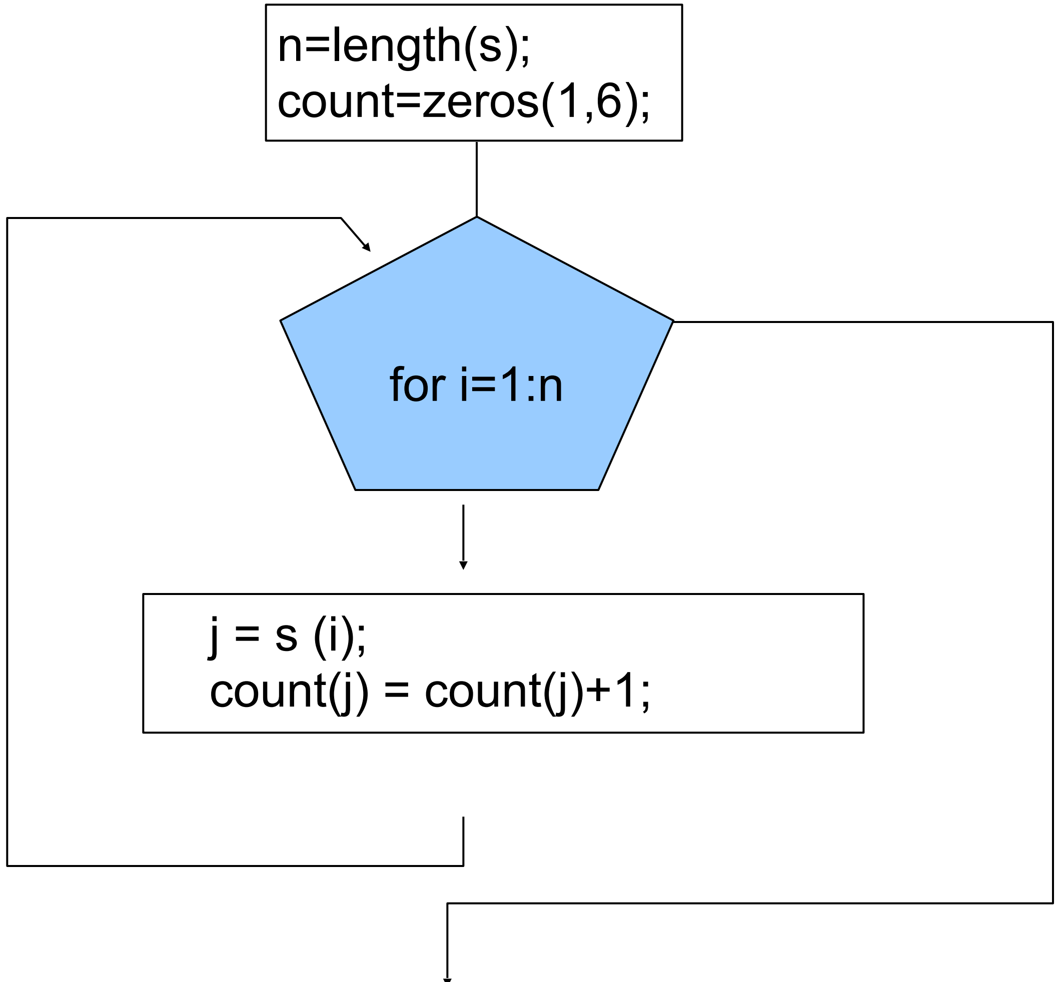
hist(s,[1 2 3 4 5 6])

Histogram

- INPUT: a series
- INPUT: [1 2 3 4 5 6]
- OUTPUT: occurrences of possible outcomes

Flow chart I :

function count=my_hist(s)

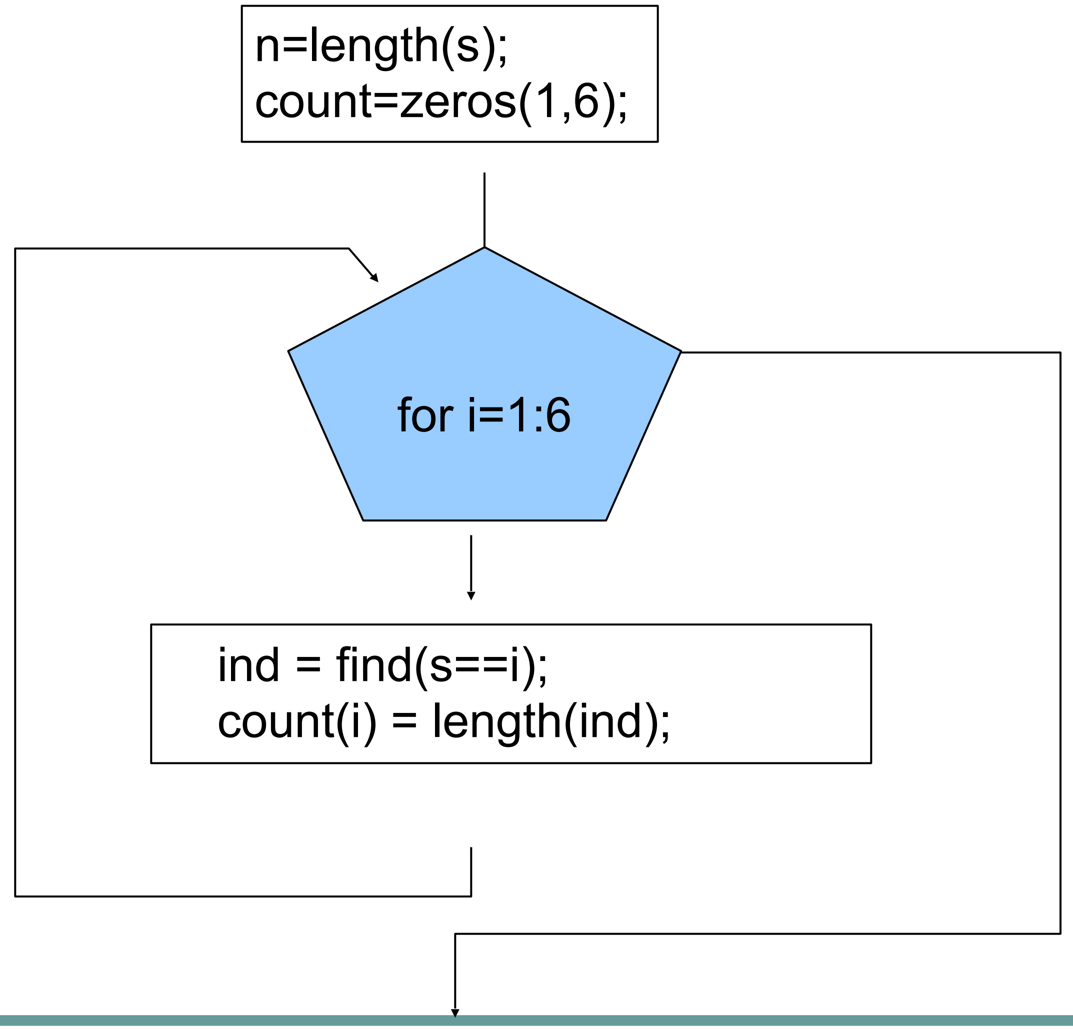


```
j=s(i);
count(j) = count(j) +1;
```

- j stores the ith elemet
- increase one for counting j

Flow chart II

function count=my_hist2(s)



function count=my_hist2(s)

```
n=length(s);
count=zeros(1,6);
for i=1:6
    ind = find(s==i);
    count(i) = length(ind);
end
```

DNA sequence

ATCG sequence

```
load mitochondria
mitochondria(1:200)
```

mitochondria(1:200)

gatcacaggtctatcaccctattaaccactcacggga
gctctccatgcatttggtattttcgtctgggggtgtgca
cgcgatagcattgcgagacgctggagccggagcac
cctatgtcgcagtatctgtctttgattcctgcctcattctatt
atttatcgcacctacgttcaatattacaggcgaacata
cctacta

ss='gatcacaggtctatcaccctattaaccact
cacgggagctctccatgcatttggtattttcgt
ctgggggtgtgcacgcgatagcattgcga
gacgctggagccggagcaccctatgtcgc
agtatctgtctttgattcctgcctcattctattattt
atcgcacctacgttcaatattacaggcgaac
atacctacta'

ATCG

- Calculate probabilities of a,t,c,g in a given ATCG sequence.

$$\Pr(x='a')=?$$

$$\Pr(x='t')=?$$

$$\Pr(x='c')=?$$

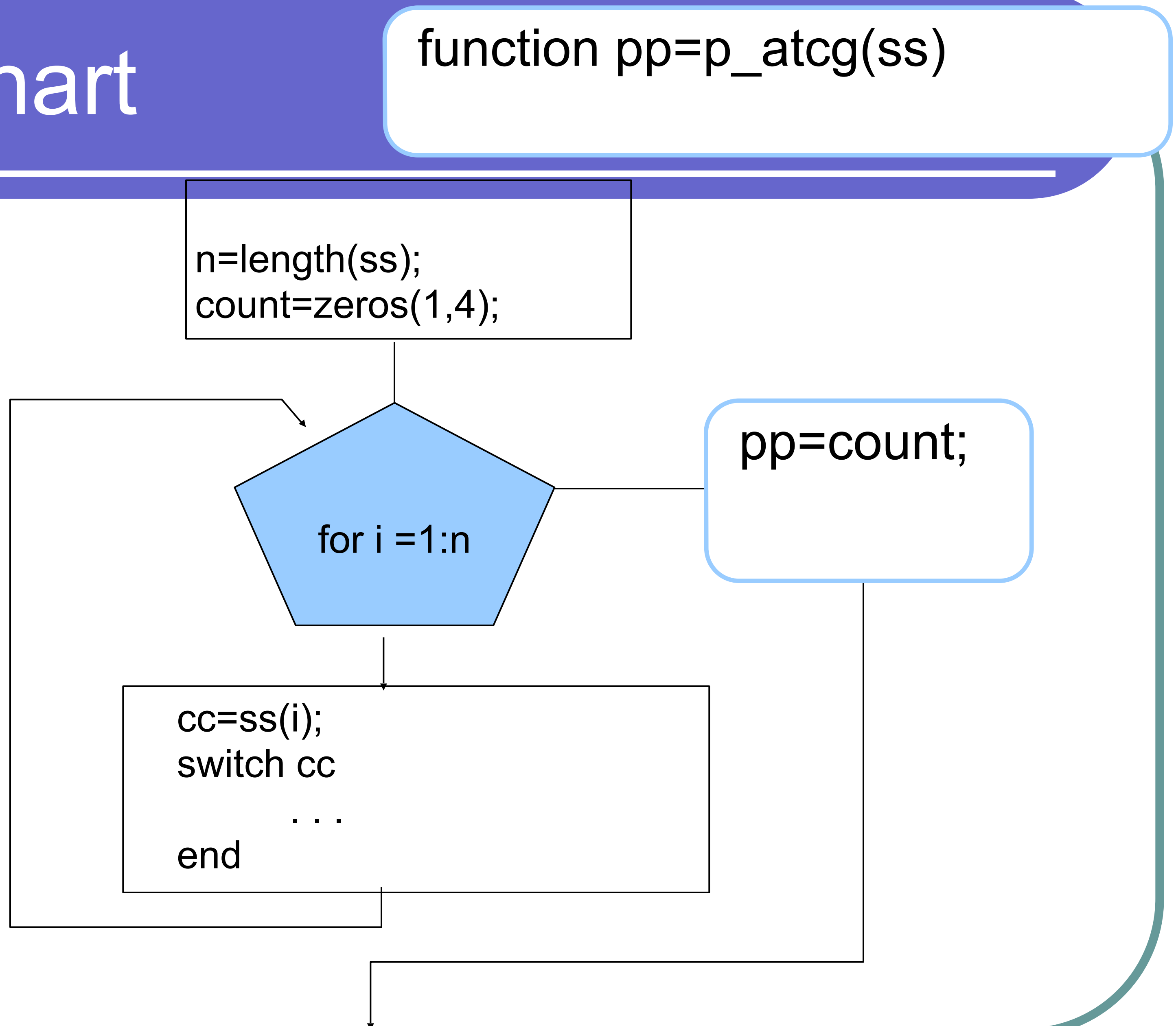
$$\Pr(x='g')=?$$

ATCG

- Count numbers of a,t,c,g in a given ATCG sequence.

Input : ss
Output : numbers of 'a', 't', 'c' and 'g'

Flow chart



function pp=p_atcg(ss)
n=length(ss);
count=zeros(1,4);
for i =1:n
pp=count;
cc=ss(i);
switch cc
. . .
end

Switch

```
cc=ss(i);
  switch cc
     case 'a'
        count(1)=count(1)+1;
     case 't'
        count(2)=count(2)+1;
     case 'c'
        count(3)=count(3)+1;
     case 'g'
        count(4)=count(4)+1;
  end
```

function pp=p_atcg(ss)

```
n=length(ss);
count=zeros(1,4);
for i =1:n
    cc=ss(i);
    switch cc
        case 'a'
            count(1)=count(1)+1;
        case 't'
            count(2)=count(2)+1;
        case 'c'
            count(3)=count(3)+1;
        case 'g'
            count(4)=count(4)+1;
    end
end
```

- INPUT: ss and s
- Count occurrences of s within ss

ss =

gtttcctact

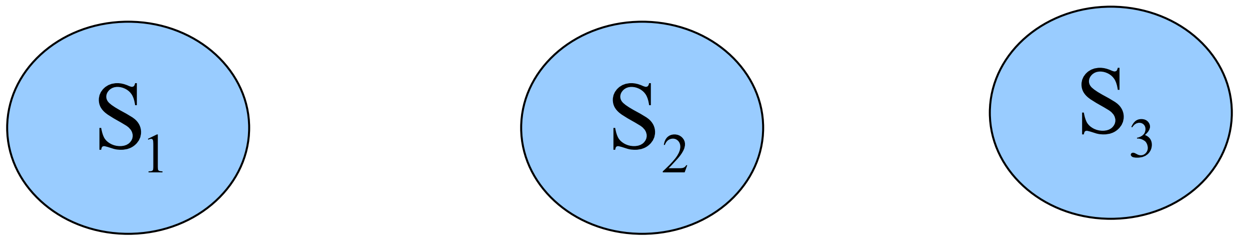
s = 'g'

```
ind=find(ss==s);
length(ind)
```

ans =

1

State



Initial population

n_i denotes the initial population of being state S_i

$$\sum_i \mathrm{n}_i = n$$

State transition

- A person at current state, e.g. S_i, may translate to one of possible states
- p_{ij} denotes the probability of state transition from S_i to S_j

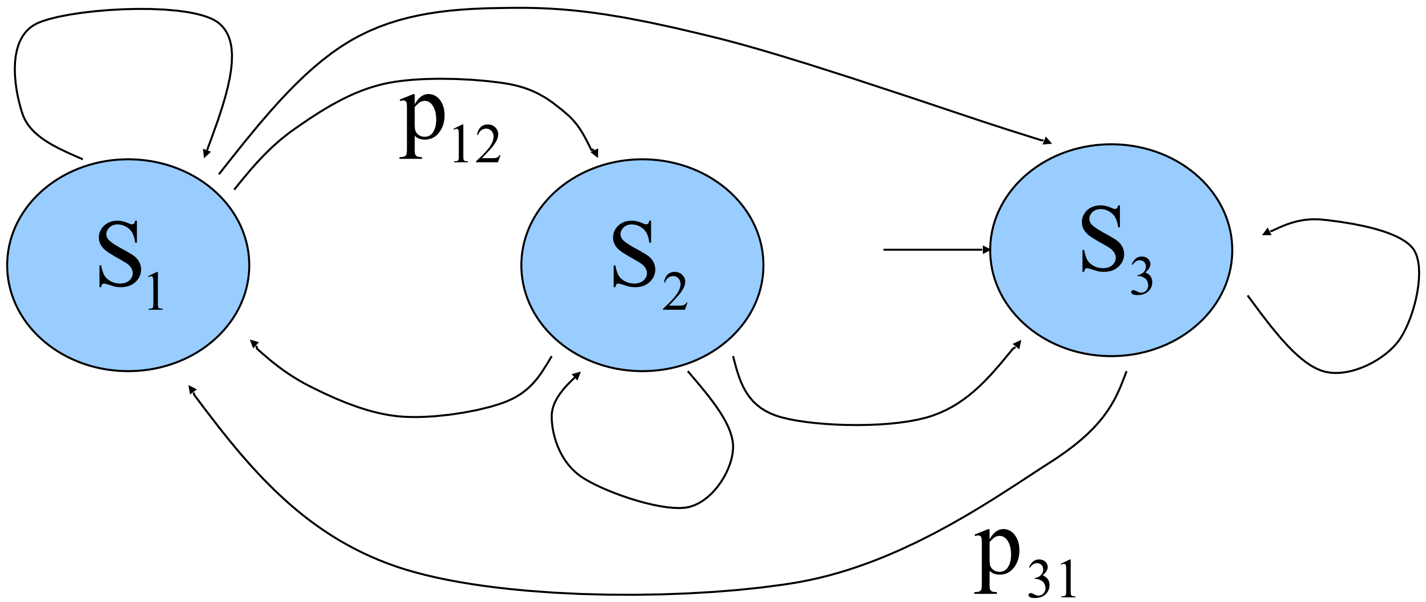
Transition probabilities

p_{ij} denotes the probability of transition from S_i to S_j

Unitary condition

$$\sum_j p_{ij} = 1$$

State



State transition

- Populations after one time of state transition

$$\mathbf{Pv}$$

$$\mathbf{P} = \begin{pmatrix} p_{11} & p_{12} & & p_{1m} \\ p_{21} & p_{22} & & p_{2m} \\ \cdots & & & \cdots \\ \cdots & & & \cdots \\ p_{m1} & p_{m2} & & p_{mm} \end{pmatrix}, \mathbf{v} = \begin{pmatrix} n_1 \\ n_2 \\ \cdots \\ n_m \end{pmatrix}$$

State transition

- Populations after state transition

$$\mathbf{P'v}$$

- Populations after k times of state transition

$$\underbrace{\mathbf{P'\ldots P'}}_{k}\mathbf{v}$$

Exercise I

- Draw a flow chart to calculate populations after k times of state transitions using for-loops
- Write a matlab function to implement the flow chart
- Find the result for

```
P =

    0.1000    0.4500    0.4500
    0.3333         0    0.6667
    0.5000         0    0.5000
```

```
v =

   100
   600
   300
```

k=1
k=10
k=20
k=100

Convergence

- Change

$$\mathbf{v}_{new} = \mathbf{P}'\mathbf{v}$$

$$\mathbf{v} - \mathbf{v}_{new}$$

- Norm

$$norm(\mathbf{v} - \mathbf{v}_{new})$$

Convergence

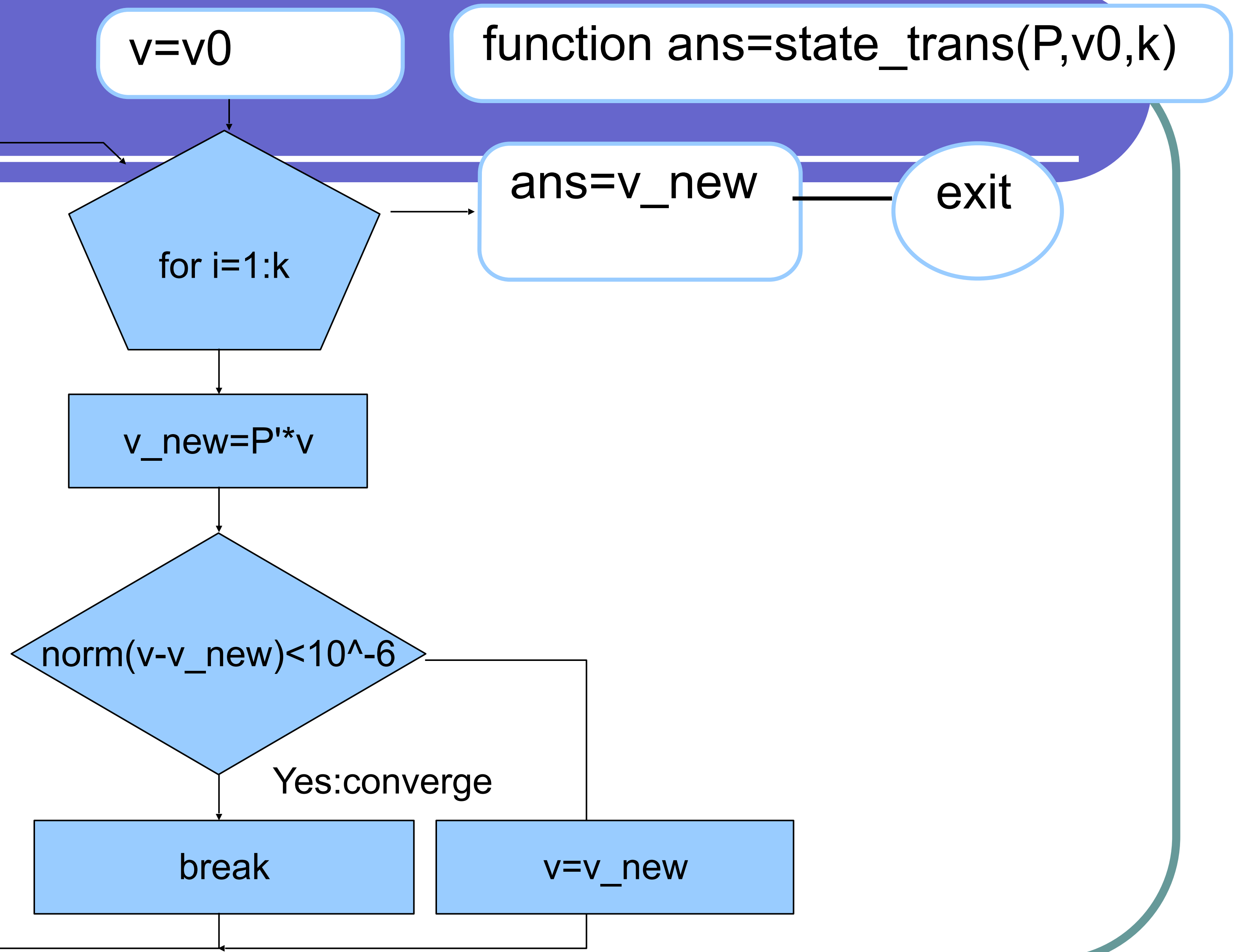
- The answer converges if

$$norm(\mathbf{v} - \mathbf{v}_{new}) \text{ is less than a small positive number}$$

e.g.

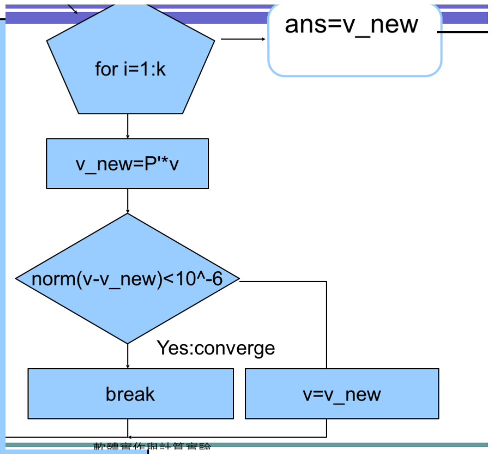
$$norm(\mathbf{v} - \mathbf{v}_{new}) < 10^{-6}$$

$$norm(\mathbf{v} - \mathbf{v}_{new}) < 10^{-8}$$


v=v0
function ans=state_trans(P,v0,k)
for i=1:k
ans=v_new
exit
v_new=P'*v
norm(v-v_new)<10^-6
Yes:converge
break
v=v_new

function ans=state_trans(P,v0,k)

```
v=v0;
for  i=1:k
   v_new=P' *v;
   if norm(v_new-v)<10^-6
      break
   else
      v=v_new
   end
end
```



$$\lim_{k\to\infty}(\mathbf{P'})^{k}\,\mathbf{v}$$

Exercise II

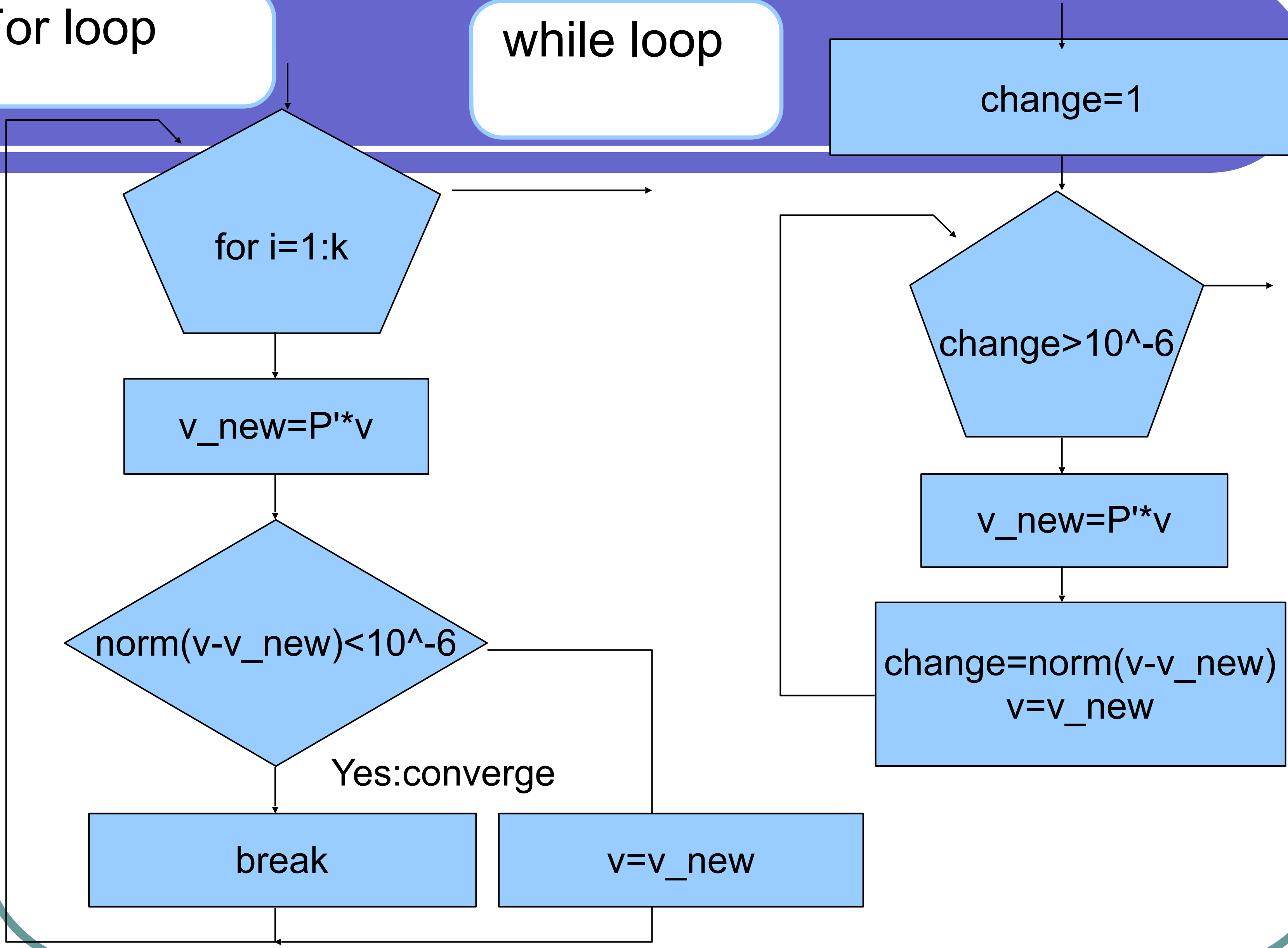
- Draw a flow chart to illustrate finding

$$\lim_{k\to\infty} (\mathbf{P}')^{k}\,\mathbf{v}$$

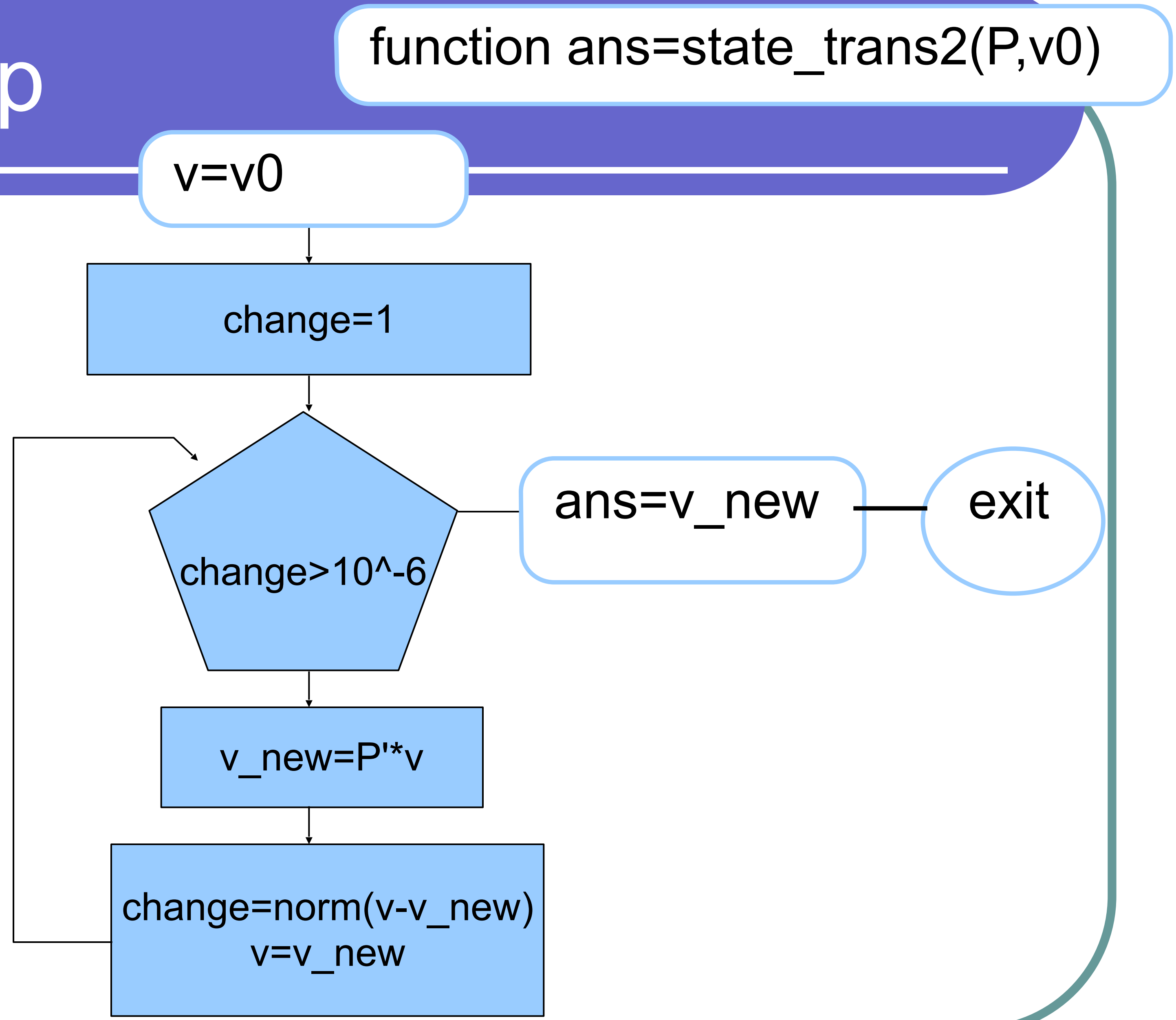
  by a for-loop
- Implement your flow chart by a matlab function

For loop

while loop

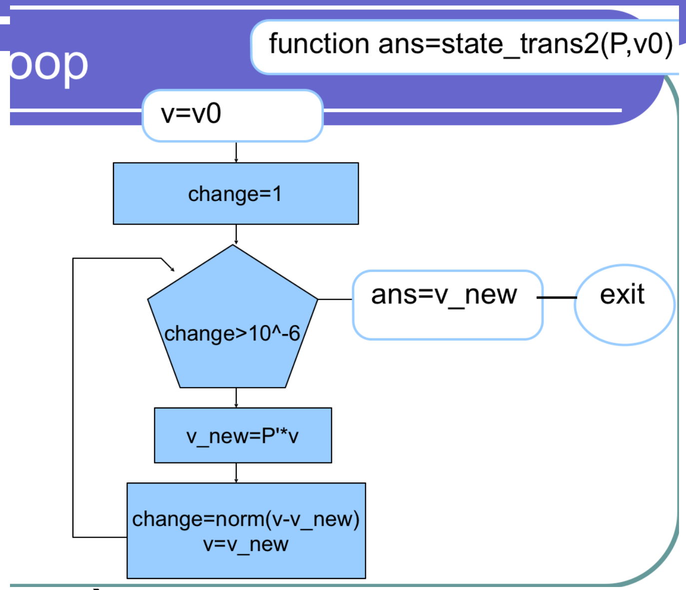


while-loop



function ans=state_trans2(P,v0)
v=v0
change=1
change>10^-6
ans=v_new
exit
v_new=P'*v
change=norm(v-v_new)
v=v_new

function ans=state_trans2(P,v0)




```
v=v0;
change = 1
while change < 10^-6
   v_new=P' *v;
   change = norm(v-v_new);
   v=v_new
end
ans=v;
```

Exercise III

- Draw a flow chart to illustrate finding

$$\lim_{k \to \infty} (\mathbf{P}')^k \mathbf{v}$$

by a while-loop

- Implement your flow chart by a matlab function