دانشگاه صنعتی خواجه نصیرالدین طوسی
دانشکده مهندسی برق- گروه مهندسی کنترل

تمرین درس محاسبات نرم

در رشتهٔ مهندسی برق گرایش مهندسی کنترل

عنوان

تمرین کلاسی دوم: بازسازی اعداد از روی اعداد مخدوش یا نویزی‌شده (مثال هگان)

نگارش

محمدجواد احمدی

استاد درس

دکتر مهدی علیاری شوره‌دلی

آذرماه ۱۴۰۱

فهرست مطالب

فهرست شکل‌ها

پاسخ سوالات

نکات قابل ذکر

کدهای متلب مربوط به این سوال از داخل پوشهٔ Codes قابل اجراست. هم‌چنین می‌توان از دستور nnd7sh در متلب و کدمنبع آورده‌شده در فهرست شکل‌ها برای شبیه‌سازی این سوال استفاده کرد.

برای حل این سوال از دستورات زیر که در پوشهٔ مربوط به کدها نیز آمده است استفاده می‌کنیم:

```
% Autoassociative network for digit recognition using Hebbian learning
clc;
clear all;
close all;

%% load training images(5x7 px)
M=7; N=5;
P=zeros(M*N,10);
for n=0:9
    s=num2str(n);
    RGB=imread(s,'png');
    m=n+1;
    P(:,m)=reshape(rgb2gray(RGB),[M*N,1]); %column prototype vector
end
P = P/255*2-1; %normalizes data to be either -1 or 1

%% compute weight matrix using Hebb rule
W=zeros(M*N,M*N);
for n=1:10
    W=W+P(:,n)*P(:,n)';
```

```
end

% simple Hebb rule did not make it recognize images well
% try again using pseudoinverse
T=P;
W1=T*pinv(P);

%% test out network with training images
for n = 10:-1:1
    a=hardlims(W*P(:,n));
    outputImg=reshape(a,[M,N]);
    figure;
    imshow(outputImg,'InitialMagnification','Fit')
end

%% test out network with noisy inputs
P_noisy = P + randi([-1,1],M*N,10);

for n = 10:-1:1
    outputImg1=reshape(P_noisy(:,n),[M,N]);
    figure;
    subplot(1,2,1)
    imshow(outputImg1,'InitialMagnification','Fit')
    title('noisy image (Hebb)')

    subplot(1,2,2)
    a=hardlims(W*P_noisy(:,n));
    outputImg2=reshape(a,[M,N]);
    imshow(outputImg2,'InitialMagnification','Fit')
    title('reconstructed image (Hebb)')
end

for n = 10:-1:1
    outputImg1=reshape(P_noisy(:,n),[M,N]);
    figure;
```

```
    subplot(1,2,1)
    imshow(outputImg1,'InitialMagnification','Fit')
    title('noisy image (Pseudoinverse)')

    subplot(1,2,2)
    a=hardlims(W1*P_noisy(:,n));
    outputImg2=reshape(a,[M,N]);
    imshow(outputImg2,'InitialMagnification','Fit')
    title('reconstructed image (Pseudoinverse)')
end

%% test out network with parts of image missing
P_partial=P;
P_partial(8:17,:)=1;

for n = 10:-1:1
    outputImg1=reshape(P_partial(:,n),[M,N]);
    figure;
    subplot(1,2,1)
    imshow(outputImg1,'InitialMagnification','Fit')
    title('partial image (Hebb)')

    subplot(1,2,2)
    a=hardlims(W*P_partial(:,n));
    outputImg2=reshape(a,[M,N]);
    imshow(outputImg2,'InitialMagnification','Fit')
    title('reconstructed image (Hebb)')
end

P_partial=P;
P_partial(8:17,:)=1;

for n = 10:-1:1
    outputImg1=reshape(P_partial(:,n),[M,N]);
    figure;
```

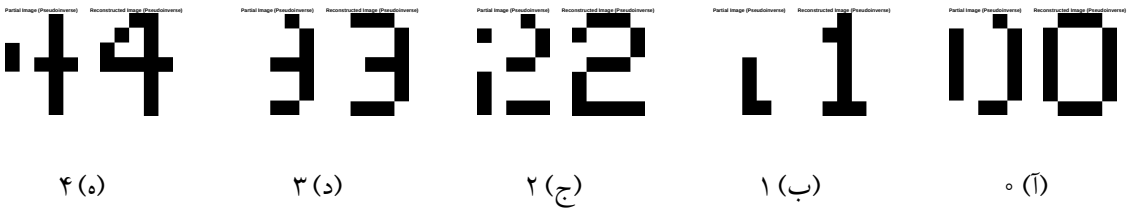
```
    subplot(1,2,1)
    imshow(outputImg1,'InitialMagnification','Fit')
    title('partial image (Pseudoinverse)')

    subplot(1,2,2)
    a=hardlims(W1*P_partial(:,n));
    outputImg2=reshape(a,[M,N]);
    imshow(outputImg2,'InitialMagnification','Fit')
    title('reconstructed image (Pseudoinverse)')
end
```

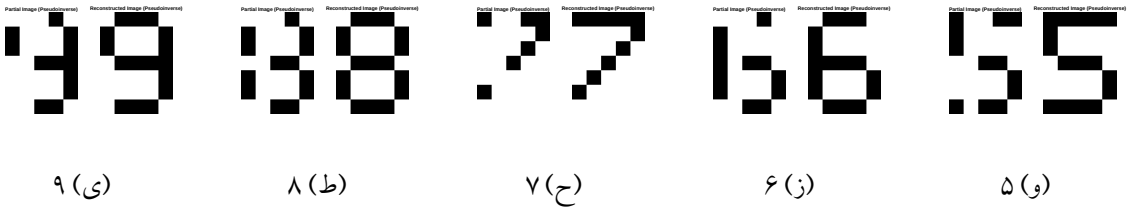
تابع hardlims هم که در کد اصلی بالا تعریف شده است به‌صورت زیر تعریف می‌گردد:

```
function [ a ] = hardlims( n )
%   hardlims Symmetric Hard Limit transfer function
%   Accepts column vector. Returns column vector.
%   behaves like unit step of amplitude 2 and vertical offset of -1
%   a = -1 for n < 0
%   a =  1 for n >= 0
    dims = size(n);
    a = zeros(dims);
    for i = 1:dims(1)
        for j = 1:dims(2)
            if n(i,j) < 0
                a(i,j) = -1;
            else
                a(i,j) = 1;
            end
        end
    end
end
```

با اجرای این دستورات نتایج به صورتی که در ادامه آورده شده است خواهد بود:

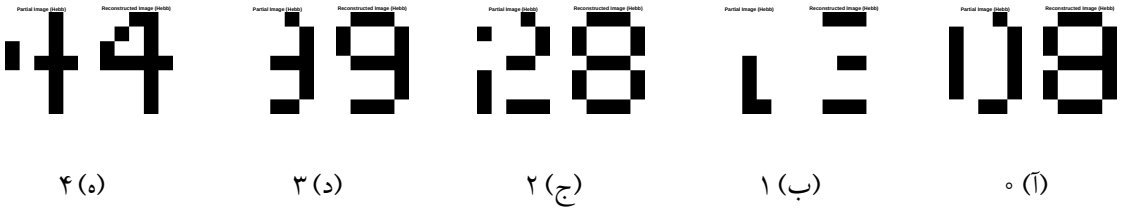


۰ (آ)　۱ (ب)　۲ (ج)　۳ (د)　۴ (ه)

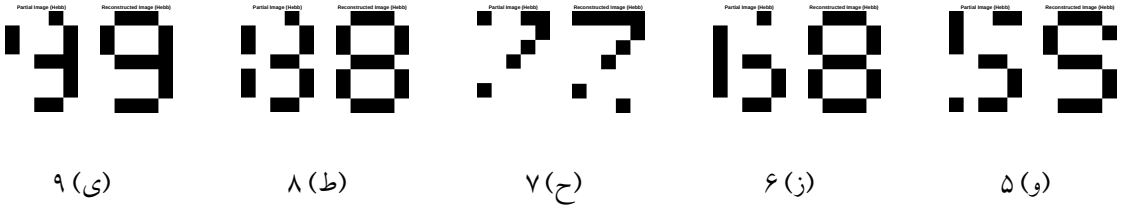


۵ (و)　۶ (ز)　۷ (ح)　۸ (ط)　۹ (ی)

شکل ۱: بازسازی اعداد از روی قسمتی از آن‌ها (Pseudoinverse).

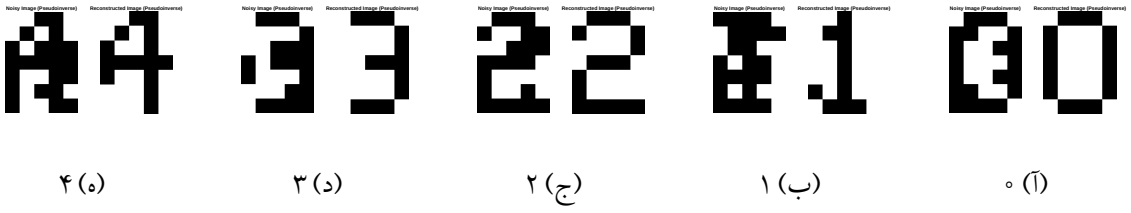


۰ (آ)　۱ (ب)　۲ (ج)　۳ (د)　۴ (ه)

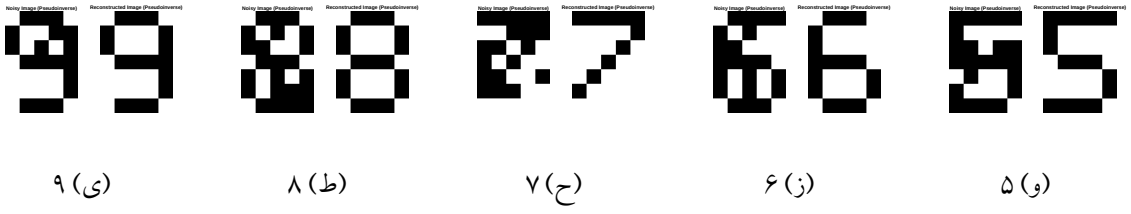


۵ (و)　۶ (ز)　۷ (ح)　۸ (ط)　۹ (ی)

شکل ۲: بازسازی اعداد از روی قسمتی از آن‌ها (Hebb).

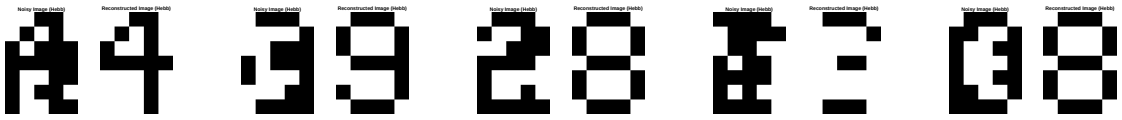


۰ (آ)　۱ (ب)　۲ (ج)　۳ (د)　۴ (ه)

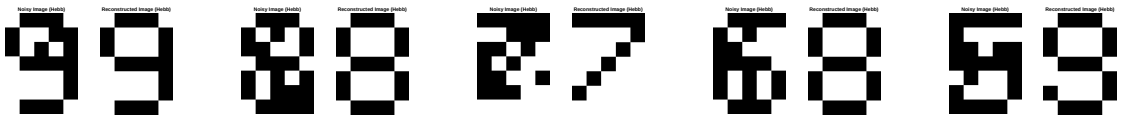


۵ (و)　۶ (ز)　۷ (ح)　۸ (ط)　۹ (ی)

شکل ۳: بازسازی اعداد از روی نویزی‌شدهٔ آن‌ها (Pseudoinverse).



۰ (آ)　　۱ (ب)　　۲ (ج)　　۳ (د)　　۴ (ه)



۵ (و)　　۶ (ز)　　۷ (ح)　　۸ (ط)　　۹ (ی)

شکل ۴: بازسازی اعداد از روی نویزی‌شدهٔ آن‌ها (Hebb).

پیوست

کدمنبع nnd7sh هم به‌صورت زیر تعریف می‌گردد:

```
function nnd7sh(cmd,arg1,arg2,arg3)
%NND7SH Supervised Hebb demonstration.

% Copyright 1994-2002 PWS Publishing Company and The MathWorks, Inc.
% $Revision: 1.6 $

%======================================================================

% CONSTANTS
me = 'nnd7sh';
p_x = 5; % pattern horizontal size
p_y = 6; % pattern vertical size

% DEFAULTS
if nargin == 0, cmd = ''; else cmd = lower(cmd); end

% FIND WINDOW IF IT EXISTS
fig = nndfgflg(me);
if length(get(fig,'children')) == 0, fig = 0; end

% GET WINDOW DATA IF IT EXISTS
if fig
  H = get(fig,'userdata');
  fig_axis = H(1);            % window axis
  desc_text = H(2);           % handle to first line of text sequence
```

```
  pattern = H(3:5);           % pattern axes 1-3
  tp_axis = H(6);             % test pattern
  rp_axis = H(7);             % response pattern
  P_ptr = H(8);               % handle to pattern matrix P
  p_ptr = H(9);               % handle to test pattern
  w_ptr = H(10);              % handle to weight matrix W
  rule1 = H(11);              % handle to first radio button
  rule2 = H(12);              % handle to second radio button
end

%=========================================================================
% Activate the window.
%
% ME() or ME('')
%=========================================================================

if strcmp(cmd,'')
  if fig
    figure(fig)
    set(fig,'visible','on')
  else
    feval(me,'init')
  end

%=========================================================================
% Close the window.
%
% ME() or ME('')
%=========================================================================

elseif strcmp(cmd,'close') & (fig)
  delete(fig)

%=========================================================================
% Initialize the window.
```

```
%
% ME('init')
%=====================================================================

elseif strcmp(cmd,'init') & (~fig)

  % STANDARD DEMO FIGURE
  fig = nndemof2(me,'DESIGN','Supervised Hebb','','Chapter 7');
  set(fig, ...
    'windowbuttondownfcn',nncallbk(me,'down'), ...
    'BackingStore','off',...
    'nextplot','add');
  H = get(fig,'userdata');
  fig_axis = H(1);
  desc_text = H(2);

  % ICON
  nndicon(7,458,363,'shadow')

  % ORIGONAL PATTERNS
  p1 = [0 1 1 1 1 0 ...
        1 0 0 0 0 1 ...
        1 0 0 0 0 1 ...
        1 0 0 0 0 1 ...
        0 1 1 1 1 0]';
  p2 = [0 0 0 0 0 0 ...
        1 0 0 0 0 0 ...
        1 1 1 1 1 1 ...
        0 0 0 0 0 0 ...
        0 0 0 0 0 0]';
  p3 = [1 0 0 0 0 0 ...
        1 0 0 1 1 1 ...
        1 0 0 1 0 1 ...
        1 0 0 1 0 1  ...
        0 1 1 0 0 1]';
```

```
P = [p1 p2 p3]*2-1;
p  = p1*2-1;

% WEIGHTS & OUTPUTS
w = P*P';
a = w*p;

% PATTERN AXES
pattern = zeros(1,5);
ltyell = nnltyell;
for k=1:5
  if k < 4
    title = sprintf('Pattern %g',k);
    pos = [25+115*(k-1) 230 100 100];
    pp = reshape(P(:,k),p_y,p_x);
    color = nngreen;
  elseif k == 4
    title = 'Test Pattern';
    pos = [25 20 160 160];
    pp = reshape(p,p_y,p_x);
    color = nndkgray;
  else
    title = 'Response Pattern';
    pos = [195 20 160 160];
    pp = reshape(a,p_y,p_x);
    color = nnred;
  end
  pattern(k) = nnsfo('a2',title,'','');
  set(pattern(k), ...
    'units','points',...
    'position',pos,...
    'color',nnltyell,...
    'xlim',[0 p_x], ...
    'ylim',[0 p_y],...
    'ydir','reverse')
```

```
    axis('off')
    pattern_h = zeros(p_y,p_x);
    box_x = [0 1 1 0 0];
    box_y = [0 0 1 1 0];
    for i=1:p_x, for j=1:p_y
      if pp(j,i) >= 0
        pattern_h(i,j) = fill(box_x+i-1,box_y+j-1,color,...
          'edgecolor',nndkblue,...
          'erasemode','none');
      else
        pattern_h(i,j) = fill(box_x+i-1,box_y+j-1,ltyell,...
          'edgecolor',nndkblue,...
          'erasemode','none');
      end
    end, end
    set(pattern(k),'userdata',pattern_h);
  end

  % WEIGHT RULE BUTTONS
  drawnow   % Let everything else appear before buttons
  rule1 = uicontrol(...
    'units','points',...
    'position',[395 190 70 20],...
    'style','radio',...
    'string','Hebb',...
    'backg',nnltgray,...
    'callback',[me '(''rule'',1)'],...
    'value',1);
  rule2 = uicontrol(...
    'units','points',...
    'position',[395 170 90 20],...
    'style','radio',...
    'string','Psuedoinverse',...
    'backg',nnltgray,...
    'max',[2],...
```

```
    'callback',[me '(''rule'',2)']);

  % BUTTONS
  if (exist('hintonw'))
    uicontrol(...
      'units','points',...
      'position',[410 140 60 20],...
      'string','Weights',...
      'callback',[me '(''weights'')'])
  end
  uicontrol(...
    'units','points',...
    'position',[410 110 60 20],...
    'string','Contents',...
    'callback','nndtoc')
  uicontrol(...
    'units','points',...
    'position',[410 80 60 20],...
    'string','Close',...
    'callback','delete(gcf)')

  % DATA POINTERS
  P_ptr = nnsfo('data'); set(P_ptr,'userdata',P);
  p_ptr = nnsfo('data'); set(p_ptr,'userdata',p);
  w_ptr = nnsfo('data'); set(w_ptr,'userdata',w);

  % SAVE WINDOW DATA AND LOCK
  H = [fig_axis desc_text pattern P_ptr p_ptr w_ptr rule1 rule2];
  set(fig,'userdata',H,'nextplot','new','color',nnltgray)

  % INSTRUCTION TEXT
  feval(me,'instr');

  nnchkfs;

```

```
%====================================================================
% Display the instructions.
%
% ME('instr')
%====================================================================

elseif strcmp(cmd,'instr') & (fig)
  nnsettxt(desc_text,...
    'Click on the green',...
    'grids to define target.',...
    'patterns. Click on the',...
    'gray grid to define',...
    'a test pattern.',...
    '',...
    'Select the rule to',...
    'calculate the network',...
    'weights below:')

%====================================================================
% Show weights.
%
% ME('weights')
%====================================================================

elseif strcmp(cmd,'weights') & (fig) & (nargin == 1)

  % GET DATA
  w = get(w_ptr,'userdata');

  f = figure;
  feval('hintonw',w);
  axis('equal');
  set(f,'name','Network Weights')
  t = get(gca,'title');
  set(t,'string','Green = Positive, Red = Negative')
```

```
236
237 %====================================================================
238 % Respond to mouse down.
239 %
240 % ME('down')
241 %====================================================================
242
243 elseif strcmp(cmd,'down') & (fig) & (nargin == 1)
244
245   set(fig,'nextplot','add')
246   for i=1:3
247     [in,x,y] = nnaxclik(pattern(i));
248     if in
249       feval(me,'down',x,y,i);
250       break
251     end
252   end
253   [in,x,y] = nnaxclik(tp_axis);
254   if in
255     feval(me,'down',x,y);
256   end
257   set(fig,'nextplot','new')
258
259 %====================================================================
260 % Respond to mouse down in pattern 1-3.
261 %
262 % ME('down',x,y,i)
263 %====================================================================
264
265 elseif strcmp(cmd,'down') & (fig) & (nargin == 4)
266
267   % GET DATA
268   x = floor(arg1)+1;
269   y = floor(arg2)+1;
270   i = arg3;
```

```
  green = nngreen;
  ltyell = nnltyell;
  red = nnred;
  P = get(P_ptr,'userdata');
  p = get(p_ptr,'userdata');
  squares = get(pattern(i),'userdata');
  rp_squares = get(rp_axis,'userdata');
  rule = get(rule2,'value')+1;

  % TOGGLE SQUARE
  ind = (x-1)*p_y+y;
  P(ind,i) = -P(ind,i);
  if P(ind,i) > 0
    set(squares(x,y),'facecolor',green);
  else
    set(squares(x,y),'facecolor',ltyell);
  end
  drawnow

  % UPDATE WEIGHTS
  if rule == 1
    w = P*P';
  else
    w = P*inv(P'*P)*P';
  end

  % STORE DATA
  set(P_ptr,'userdata',P);
  set(w_ptr,'userdata',w);

  % UPDATE OUTPUTS
  feval(me,'update')

%=======================================================================
% Respond to mouse down in test pattern.
```

```
%
% ME('down',x,y)
%=====================================================================

elseif strcmp(cmd,'down') & (fig) & (nargin == 3)

  % GET DATA
  x = floor(arg1)+1;
  y = floor(arg2)+1;
  dkgray = nndkgray;
  ltyell = nnltyell;
  tp_squares = get(tp_axis,'userdata');
  rp_squares = get(rp_axis,'userdata');
  p = get(p_ptr,'userdata');
  w = get(w_ptr,'userdata');

  % TOGGLE SQUARE
  ind = (x-1)*p_y+y;
  p(ind) = -p(ind);
  if p(ind) > 0
    set(tp_squares(x,y),'facecolor',dkgray);
  else
    set(tp_squares(x,y),'facecolor',ltyell);
  end
  drawnow

  % STORE DATA
  set(p_ptr,'userdata',p);

  % UPDATE OUTPUTS
  feval(me,'update')

%=====================================================================
% Set weight rule.
%
```

```
% ME('rule',i)
%=============================================================

elseif strcmp(cmd,'rule') & (fig) & (nargin == 2)

  % SET RADIO BUTTONS
  if arg1 == 1
    set(rule1,'value',1)
    set(rule2,'value',0)
  else
    set(rule1,'value',0)
    set(rule2,'value',2)
  end

  % GET DATA
  P = get(P_ptr,'userdata');

  % UPDATE WEIGHTS
  if arg1 == 1
    w = P*P';
  else
    w = P*inv(P'*P)*P';
  end

  % STORE DATA
  set(w_ptr,'userdata',w)

  % UPDATE OUTPUTS
  feval(me,'update')

%=============================================================
% Update response pattern.
%
% ME('update')
%=============================================================
```

```
376
377 elseif strcmp(cmd,'update') & (fig)
378
379   % GET DATA
380   red = nnred;
381   ltyell = nnltyell;
382   p = get(p_ptr,'userdata');
383   w = get(w_ptr,'userdata');
384   rp_squares = get(rp_axis,'userdata');
385
386   % UPDATE OUTPUTS
387   a = w*p;
388   a = reshape(a,p_y,p_x);
389   for i=1:p_x, for j=1:p_y
390     if a(j,i) > 0
391       set(rp_squares(i,j),'facecolor',red)
392     else
393       set(rp_squares(i,j),'facecolor',ltyell)
394     end
395   end, end
396
397 %=======================================================================
398 end
```