Electronic Supplementary Material (ESI) for RSC Advances.

Execution script NRdensityfun.m

```
% Large area quantitative analysis of nanostructured thin-films
% Rafal Sliz, Chibuzor Eneh, Yuji Suzuki, Jakub Czajkowski, Tapio Fabritius, Poopathy
Kathirgamanathan, Arokia Nathan, Risto Myllyla and Ghassan E. Jabbour


opengl software
tic,

close all, clc, clear all
h = waitbar(0,'Loading images.....');

%%%%%%%%%%%%%%%%%%%%%%% PREPARE FILE INPUT %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
imgPath = '\\samba.ee\kuba\ZnO Nanorods\warvick'; dCell = dir(fullfile(imgPath,'*.jpg')); %
Please provide file extension
feature_name='nanorods'; % Please provide name of studied features
make_movies=0; % Should movies be saved?

%%%%%%%%%%%%%%%%%%%% SORT FILE NAMES BY CREATION TIME %%%%%%%%%%%%%%%%%%%%%
[c r]=size(dCell);
for i=1:c
    dates(i,:)=dCell(i,1).datenum;
end
[dates indexes]=sort(dates);

%%%%%%%%%%%%%%%%%%%%%%% READ IMAGE FILES FROM DISK %%%%%%%%%%%%%%%%%%%%%%%%%
for a=1:c
    waitbar(a/c)
    I(:,:,a) = imread(fullfile(imgPath,dCell(indexes(a),1).name));
    sorted_dates(a,:)=dCell(indexes(a),1).date;
end

if (isempty(I))
    disp('No files were selected')
end

%%%%%%%%%%%%%%%%%%%%%%% PROCESS AND ANALYSE IMAGES %%%%%%%%%%%%%%%%%%%%%%%%%
if make_movies == 1
    aviobj2 = avifile('all_artefacts.avi','fps',1);
    h = waitbar(0,'Processing images.....');
    for i = 1:c;

[how_many_nrd(i,:),nrd_occupied_area_in_pixels(i,:),area_of_nanorods_physical(i,:),areal_distrib(
i,:),processed_image(i,:,:),aviobj2]=NRdensityfun1_3_comments(I(:,:,i),aviobj2,feature_name);
    end
    aviobj2 = close(aviobj2);
else
    h = waitbar(0,'Processing images.....');
    for i = 1:c;

[how_many_nrd(i,:),nrd_occupied_area_in_pixels(i,:),area_of_nanorods_physical(i,:),areal_distrib(
i,:),processed_image(i,:,:),aviobj2]=NRdensityfun1_3_comments(I(:,:,i),0,feature_name);
    end
end
close(h)
toc;
```

Main processing function NRdensityfun.m

```
function
[how_many_nrd,nrd_occupied_area_in_pixels,area_of_nanorods_physical,areal_distrib,processed_image
,aviobj2] = NRdensityfun1_3_comments(source_image,aviobj2,feature_name)
%%
clc,
opengl software % compatibility setting
```

```
[r c] = size(source_image);
font_scale=1.5; % scale captions
font_size=ceil(0.0175*r*font_scale);
text_x=ceil(0.015*c); % position text
text_y=ceil(0.94*r); % position text

wiener_kernel_scale=1; % kernel scale for Wiener filter
gaussian_kernel_scale=1; % kernel scale for Gaussian filter
wiener_threshold=0.95; % threshold for artefacts in Wiener filter result
gaussian_threshold=0.05; % threshold for artefacts in Gaussian filter result
otsu_threshold=.8; % threshold binarization
dark_artefact_area_scale=3; % threshold for artefacts' size
lighting_scale=0.25; % lighting correction strength
caption_crop_scale=0.15; % how much of the height is occupied by SEM caption
image_scale=100/49; % set the scale for the SEM image [nm/px]
sprintf('Nanos per pixel: %0.f',image_scale) %show computed scale
max_feature_radius=20; % expected maximum size of features
min_feature_radius=3; % expected minimum size of features

cmap=colorcube(max_feature_radius); % make color map

if aviobj2 ~=0
    aviobj = avifile('mymovie.avi','fps',0.5); % prepare movie file
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% FIGURE 1 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if aviobj2 ~=0
    figure;imshow(mat2gray(source_image));
    text(text_x,text_y,'\color{white}1: Original SEM image','BackgroundColor',[0 0 0],
'FontSize',font_size)
    aviobj = addframe(aviobj,getframe);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Crop SEM image %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

cropped_image = imcrop(source_image,[0 0 c ceil(r-caption_crop_scale*r)]);

hist_low_high=stretchlim(cropped_image);
hist_low_high(1)=1.05*hist_low_high(1);
hist_low_high(2)=0.95*hist_low_high(2);
cropped_image = imadjust(cropped_image,hist_low_high,[]);
[Ir Ic] = size(cropped_image);
cropped_space=zeros(r-Ir,c);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% FIGURE 2 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if aviobj2 ~=0
    figure,imshow(add_cropped_space(cropped_image,cropped_space));
    text(text_x,text_y,'\color{white}2: Caption is cropped','BackgroundColor',[0 0 0],
'FontSize',font_size)
    aviobj = addframe(aviobj,getframe);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%% Define padding parameters %%%%%%%%%%%%%%%%%%%%%%%%%
imasize=size(cropped_image);
start_x=1;
stop_x=imasize(2);
start_y=1;
stop_y=imasize(1);
add_offset=128;
padding_size=round(imasize(2)/4);

%%%%%%%%%%%%%%%%%%%% Make kernels for Gaussian filters %%%%%%%%%%%%%%%%%%%%%
xg=linspace(-3,3,ceil(gaussian_kernel_scale*16));
yg=xg';
```

```
[Xg,Yg]=meshgrid(xg,yg);
gau_kernel=exp(-(Xg.^2+Yg.^2)/2);

xg2=linspace(-3,3,ceil(gaussian_kernel_scale*32));
yg2=xg2';
[Xg2,Yg2]=meshgrid(xg2,yg2);
gau2_kernel=exp(-(Xg2.^2+Yg2.^2)/2);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Pad the image %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
padded_image=single(squeeze(mean(cropped_image,3))+add_offset);
padded_image=padded_image(start_y:stop_y,start_x:stop_x);
padded_image=padarray(padded_image,[padding_size padding_size],'replicate');

%%%%%%%%%%%%%%%%%%%%%%%%%%%% Correct illumination %%%%%%%%%%%%%%%%%%%%%%%%%%%%
Ig2=mat2gray(conv2(padded_image,gau2_kernel,'same'));
gaus2=1-
mat2gray(Ig2(padding_size+1:imasize(1)+padding_size,padding_size+1:imasize(2)+padding_size));
cropped_image=mat2gray(mat2gray(cropped_image)+lighting_scale*gaus2);
clear padded_image

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Pad the image %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
padded_image=single(squeeze(mean(cropped_image,3))+add_offset);
padded_image=padded_image(start_y:stop_y,start_x:stop_x);
padded_image=padarray(padded_image,[padding_size padding_size],'replicate');

%%%%%%%%%%%%%%%%%%%%%%%%%%%% Use Wiener filter %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[mb, nb]=bestblk(size(padded_image));
wienered = wiener2(padded_image,ceil(wiener_kernel_scale*[mb nb]));
wiener_image=wienered(padding_size+1:imasize(1)+padding_size,padding_size+1:imasize(2)+padding_si
ze);
wiener_image=mat2gray(wiener_image);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% FIGURE 3 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if aviobj2 ~=0
    figure;imshow(add_cropped_space(wiener_image,cropped_space));
    text(text_x,text_y,'\color{white}3: Wiener filtering to identify bright
artefacts','BackgroundColor',[0 0 0], 'FontSize',font_size)
    aviobj = addframe(aviobj,getframe);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%% Determine bright artefacts %%%%%%%%%%%%%%%%%%%%%%%%%
min_wien=min(min((wiener_image)));
max_wien=max(max((wiener_image)));
wiener_lvl=min_wien+wiener_threshold*(max_wien-min_wien);
bright_mask = im2bw(mat2gray(wiener_image),wiener_lvl);
se = strel('disk',3);
bright_mask = imdilate(bright_mask,se);
bright_mask = bwareaopen(bright_mask, 300);
cropped_wo_bright=mat2gray(cropped_image);
cropped_wo_bright(bright_mask>0)=0.1;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% FIGURE 4 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if aviobj2 ~=0
    figure;imshow(add_cropped_space(bright_mask,cropped_space));
    text(text_x,text_y,'\color{white}4: Bright artefact mask is created','BackgroundColor',[0 0
0], 'FontSize',font_size)
    aviobj = addframe(aviobj,getframe);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Make RGB mask %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
temp=add_cropped_space(bright_mask,cropped_space);
rgbMask=make_RGB_mask(255,0,0,temp);
temp=uint8(cat(1,255.*mat2gray(cropped_image),255.*cropped_space));
rgbImage =cat(3,temp,temp,temp);
temp=add_cropped_space(bright_mask,cropped_space);
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% FIGURE 5 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if aviobj2 ~=0
    figure;imshow(rgbImage);
    hold on
    hh = imshow(rgbMask); % Save the handle; we'll need it later
    hold off
    set(hh, 'AlphaData', 0.5.*temp)
    text(text_x,text_y,'\color{white}5: SEM image with identified bright
artefacts','BackgroundColor',[0 0 0], 'FontSize',font_size)
    hold off
    aviobj = addframe(aviobj,getframe);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% FIGURE 6 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if aviobj2 ~=0
    figure;imshow(add_cropped_space(cropped_wo_bright,cropped_space));
    text(text_x,text_y,'\color{white}6: SEM image - bright artefacts are
deleted','BackgroundColor',[0 0 0], 'FontSize',font_size)
    aviobj = addframe(aviobj,getframe);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Padd the image %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
cropped_wo_bright2=cropped_wo_bright;
for i=1:5
    cropped_wo_bright2=imdilate(cropped_wo_bright2,strel('disk',1,8));
end
padded_image=single(squeeze(mean(mat2gray(cropped_wo_bright2),3))+add_offset);
padded_image=padded_image(start_y:stop_y,start_x:stop_x);
padded_image=padarray(padded_image,[padding_size padding_size],'replicate');

%%%%%%%%%%%%%%%%%%%%%%%% Use Gaussian filter %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Ig=conv2(padded_image,gau_kernel,'same');
dark_mask_negative=Ig(padding_size+1:imasize(1)+padding_size,padding_size+1:imasize(2)+padding_si
ze);
cropped_wo_bright=mat2gray(double(cropped_wo_bright).*double(dark_mask_negative));
clear padded_image

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% FIGURE 7 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if aviobj2 ~=0
    figure;imshow(add_cropped_space(dark_mask_negative,cropped_space));
    text(text_x,text_y,'\color{white}7: The image is Gaussian filtered','BackgroundColor',[0 0
0], 'FontSize',font_size)
    aviobj = addframe(aviobj,getframe);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%% Determine dark artefacts %%%%%%%%%%%%%%%%%%%%%%%%%%%
amed=median(median(mat2gray(dark_mask_negative)));
dark_mask=~(im2bw(mat2gray(dark_mask_negative),gaussian_threshold*amed));
dark_mask = bwareaopen(dark_mask, floor(5760/2/dark_artefact_area_scale),8);
filled = imfill(dark_mask, 'holes');
holes = filled & ~dark_mask;
bigholes = bwareaopen(holes, floor(1600/dark_artefact_area_scale));
smallholes = holes & ~bigholes;
dark_mask = dark_mask | smallholes;
rozmiar_ciemne=sum(nnz(dark_mask));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% FIGURE 8 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if aviobj2 ~=0
    figure;imshow(add_cropped_space(dark_mask,cropped_space));
    text(text_x,text_y,'\color{white}8: Mask for significant dark artefacts is
computed','BackgroundColor',[0 0 0], 'FontSize',font_size)
    aviobj = addframe(aviobj,getframe);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Make RGB mask %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
temp2=add_cropped_space(dark_mask,cropped_space);
rgbMask2=make_RGB_mask(0,255,0,temp2);
temp2=uint8(cat(1,255.*mat2gray(cropped_wo_bright),255.*cropped_space));
rgbImage2 =cat(3,temp2,temp2,temp2);
temp2=add_cropped_space(dark_mask,cropped_space);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% FIGURE 9 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if aviobj2 ~=0
    figure;imshow(rgbImage2);
    hold on
    hh = imshow(rgbMask2); % Save the handle; we'll need it later
    hold off
    set(hh, 'AlphaData', 0.5.*temp2)
    text(text_x,text_y,'\color{white}9: SEM image with identified dark
artefacts','BackgroundColor',[0 0 0], 'FontSize',font_size)
    aviobj = addframe(aviobj,getframe);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%% Check size of dark regions %%%%%%%%%%%%%%%%%%%%%%%%%%
[y x]=size(cropped_image);
if (rozmiar_ciemne<0.7*(x*y))
    processed_image=cropped_wo_bright;
    processed_image(dark_mask>0)=0.1;
else
    processed_image=cropped_wo_bright;
    dark_mask=0*dark_mask;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% FIGURE 10 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if aviobj2 ~=0
    figure;imshow(add_cropped_space(cropped_image,cropped_space));
    hold on
    hh = imshow(rgbMask); % Save the handle
    hh2 = imshow(rgbMask2); % Save the handle
    hold off
    set(hh, 'AlphaData', 0.5.*temp)
    set(hh2, 'AlphaData', 0.5.*temp2)
    text(text_x,text_y,'\color{white}10: SEM with marked all artefacts','BackgroundColor',[0 0
0], 'FontSize',font_size)
    aviobj = addframe(aviobj,getframe);
    aviobj2 = addframe(aviobj2,getframe);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% FIGURE 11 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if aviobj2 ~=0
    figure;imshow(add_cropped_space(processed_image,cropped_space));
    text(text_x,text_y,'\color{white}11: Final processed image','BackgroundColor',[0 0 0],
'FontSize',font_size)
    aviobj = addframe(aviobj,getframe);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%% Prepare the processed image for segmentation %%%%%%%%%%%%%%%
hist_low_high=stretchlim(processed_image);
hist_low_high(1)=1.05*hist_low_high(1);
hist_low_high(2)=0.95*hist_low_high(2);
processed_image = imadjust(processed_image,hist_low_high,[]);

full_mask=im2bw(dark_mask+bright_mask,0.1);
total_artefacts=sum(nnz(full_mask));

processed_image=mat2gray(processed_image);
```

```
gray_lvl=graythresh(processed_image);
substr=im2bw(processed_image,otsu_threshold*gray_lvl);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% FIGURE 12 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%% NANOROD COUNTING AND SIZE ALLOCATION %%%%%%%%%%%%%%%%%%%%%%
if aviobj2 ~=0
    hold off
    figure,imshow(add_cropped_space(processed_image,cropped_space));
    hold on
end
% preallocate variables and parameters
[x y]=size(add_cropped_space(processed_image,cropped_space));
mxr=max_feature_radius;
mnr=min_feature_radius;
area_temp=zeros(x,y,mxr-2);
rgbNanorods=zeros(x,y,3,mxr-2);
how_many_nrd=zeros(mxr-2,1);
nrd_occupied_area_in_pixels=zeros(mxr-2,1);
area_of_nanorods_physical=zeros(mxr-2,1);

% segmentation and counting loop
for i=1:mxr-2
    if (mxr-1-i)>mnr
        s=strel('disk',mxr-1-i,0);
        z=imopen(substr,s);
        area_temp(:,:,mxr-i)=add_cropped_space(z,cropped_space);
        temp_mask=area_temp(:,:,mxr-i);
        rgbNanorods(:,:,:,mxr-i)=make_RGB_nanorods_mask(temp_mask,cmap,i+1);
        imshow(add_cropped_space(processed_image,cropped_space))
        count=mxr-1-i;
        current_feature_size=image_scale*count;
        CC=bwconncomp(z,6);
        how_many_nrd(mxr-i)=CC.NumObjects;
        nrd_occupied_area_in_pixels(mxr-i)=nnz(z);
        area_of_nanorods_physical(mxr-i)=pi*current_feature_size^2*CC.NumObjects;
        substr=imsubtract(substr,z);
        if aviobj2 ~=0
            hhh=imshow(rgbNanorods(:,:,:,mxr-i));
            set(hhh, 'AlphaData', 0.5.*area_temp(:,:,mxr-i))
            text(text_x,text_y,sprintf('\\color{white}12: Morphology-based segmentation.\n
Counting %s of the %.0f nm diameter. Found: %d', feature_name, 2*current_feature_size,
how_many_nrd(max_feature_radius-i)),'BackgroundColor',[0 0 0], 'FontSize',font_size)
            aviobj = addframe(aviobj,getframe);
        end
    else
        area_temp(:,:,mxr-i)=add_cropped_space(zeros(size(z)),cropped_space);
        temp_mask=area_temp(:,:,mxr-i);
        rgbNanorods(:,:,:,mxr-i)=make_RGB_nanorods_mask(temp_mask,cmap,1);
    end
end

% Take into account unallocated features
CC=bwconncomp(substr,6);
how_many_nrd(1)=CC.NumObjects;
nrd_occupied_area_in_pixels(1)=nnz(substr);
area_of_nanorods_physical(1)=nnz(substr)*pi;

area_temp(:,:,1)=add_cropped_space(zeros(size(z)),cropped_space);
temp_mask=area_temp(:,:,1);
rgbNanorods(:,:,:,1)=make_RGB_nanorods_mask(temp_mask,cmap,1);

% Display final image with all of the identified features
if aviobj2 ~=0
    imshow(add_cropped_space(cropped_image,cropped_space));
    hold on

    for i=1:mxr-1
```

```
        hhh=imshow(rgbNanorods(:,:,:,i));
        set(hhh, 'AlphaData', 0.5.*area_temp(:,:,i))
    end
    hold off
    text(text_x,text_y,sprintf('\\color{white}12: Morphology-based segmentation.\n Total number
of %s found in the image: %d',feature_name,sum(how_many_nrd(4:end))),'BackgroundColor',[0 0 0],
'FontSize',font_size)
    aviobj = addframe(aviobj,getframe);
    aviobj2 = addframe(aviobj2,getframe);
end

% Determine areal distribution
areal_distrib=sum(nrd_occupied_area_in_pixels(2:end))/((x*y)-total_artefacts-
nrd_occupied_area_in_pixels(1))*100;
if aviobj2 ~=0
    aviobj=close(aviobj);
end

end
```

add_cropped_space.m

```
function Io=add_cropped_space(I,cropped_space)
% FUNCTION FILLS THE CROPPED IMAGE WITH A BLACK SPACE TO ITS ORIGINAL SIZE
Io=cat(1,mat2gray(I),cropped_space);
end
```

make_RGB_mask.m

```
function rgbMask=make_RGB_mask(g,r,b,temp);
%FUNCTION MAKES AN RGB MASK BASED ON INPUT ARRAY AND COLOR VALUE
ggg=uint8(g.*temp);
rrr=uint8(r.*temp);
bbb=uint8(b.*temp);
rgbMask = cat(3,rrr,ggg,bbb);
end
```

make_RGB_nanorods_mask.m

```
function rgbNanoMask=make_RGB_nanorods_mask(temp_mask,cmap,index);
% FUNCTION CREATES RGB IMAGE BASED ON MASK ARRAY AND COLOR MAP INDEX
rrz=uint8(255.*temp_mask.*cmap(index,1));
ggz=uint8(255.*temp_mask.*cmap(index,2));
bbz=uint8(255.*temp_mask.*cmap(index,3));
rgbNanoMask=cat(3,rrz,ggz,bbz);
end
```