PSYCHOPHYSICAL EXPERIMENTS

A LETTER MATCHING TASK

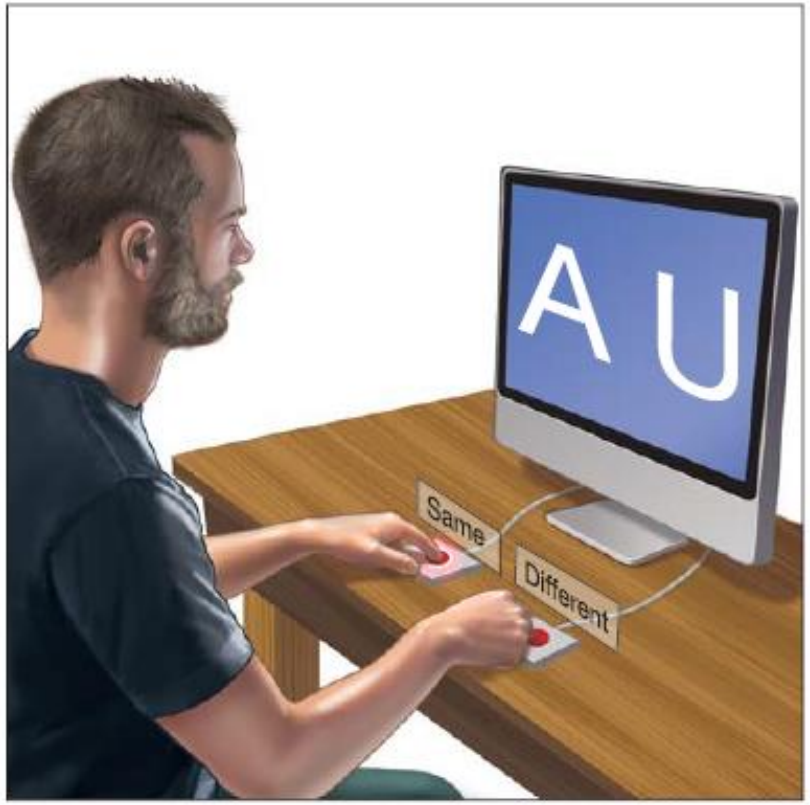
POSNER

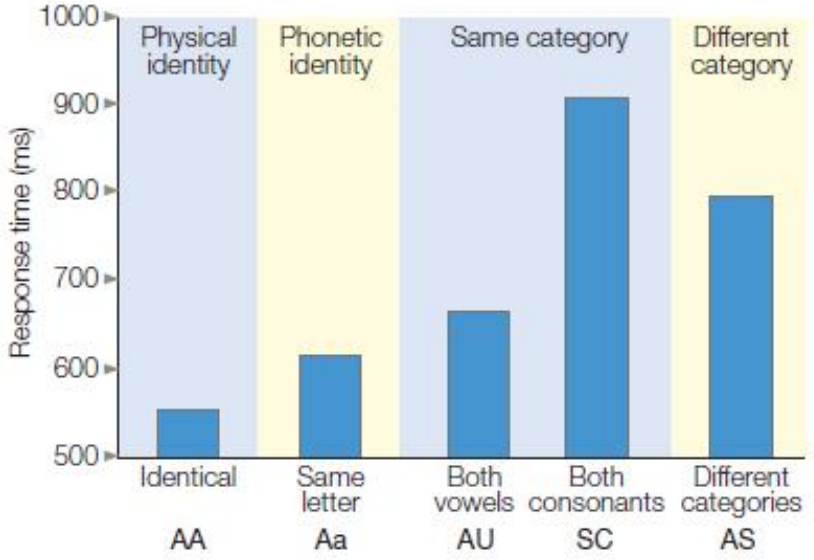
ocacdrngi ot a sehrerearc ta maccbriegd ineyurvtis, ti edost'n rttaem ni awth rreod eht tlteser ni a rwdo rea, eht ylon pirmtoatn gihtn si atth het rifts nda satl ttelre eb tat het ghitr clepa. eht srte anc eb a otlta sesm dan ouy anc itlls arde ti owtuthi monrbel ihst si cebusea eth n yb stifle, tub eth

Aoccdrnig to a rseheearcr at Cmabrigde Uinervtisy, it deosn't mttaer in waht oredr the ltteers in a wrod are, the olny iprmoatnt tihng is taht the frist and lsat ltteer be at the rghit pclae. The rset can be a total mses and you can sitll raed it wouthit porbelm. Tihs is bcuseae the huamn mnid deos not raed ervey lteter by istlef, but the wrod as a wlohe.

MENTAL REPRESENTATION

- A letter-matching task (Posner) – Chronometric (reaction time task)
- Reaction time (RT) of subject is the dependent variable.
- Response time can show how fast each representation is activated
- q) Why 'both consonants' condition takes longer for response than 'both vowels' condition?





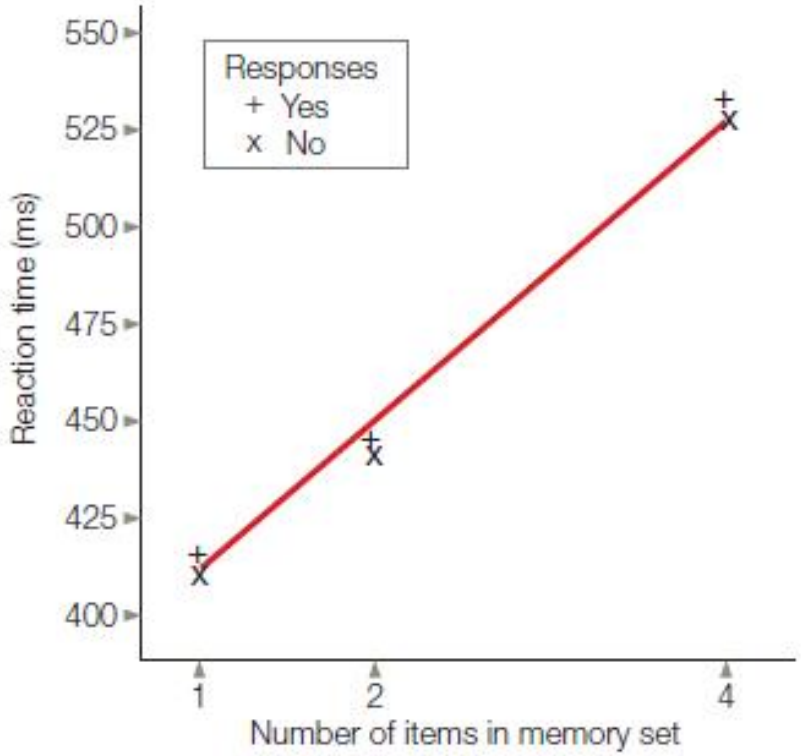
MEMORY COMPARISON TASK

(MEMORY-SCANNING TASK)

SAUL STERNBERG

REPRESENTATIONS ARE TRANSFORMED

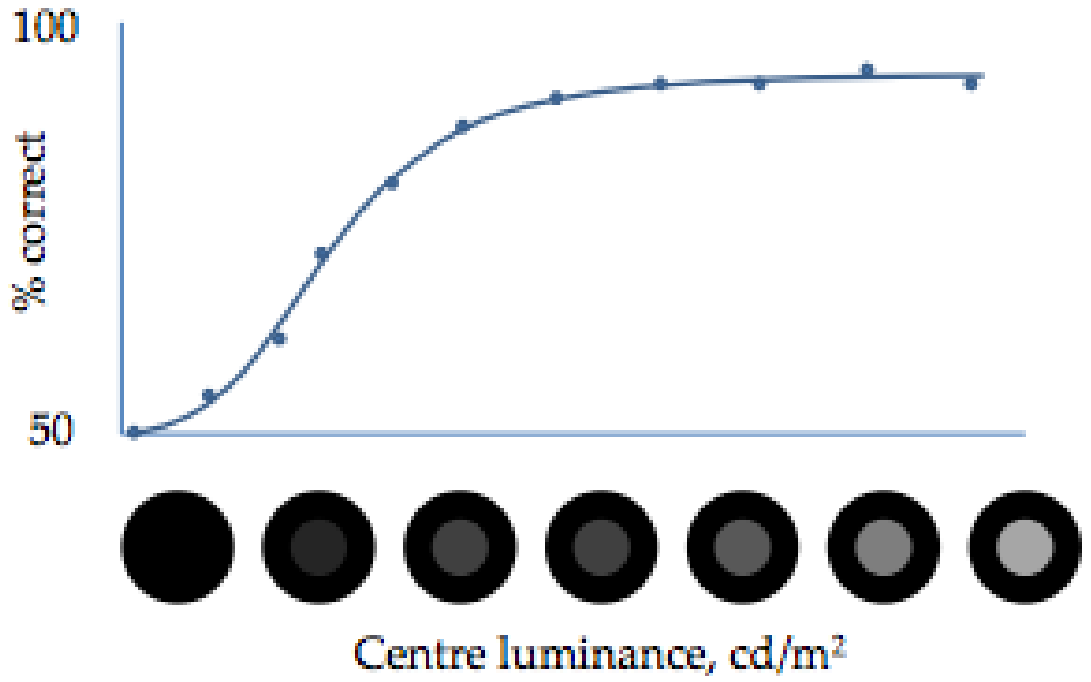
- Saul Sternberg (1975) Memory comparison task
  - Exhausting vs. self-terminating search



WORD SUPERIORITY EFFECT

REICHER

TWO ALTERNATIVE FORCED CHOICE PARADIGM



PARALLEL VS. SERIAL PROCESSES

Reicher (1969) Word superiority effect

Does the stimulus contain an A or an E?

Condition	Stimulus	Accuracy
Word	RACK	90%
Nonsense string	KARC	80%
Xs	XAXX	80%

Two alternative forced choice

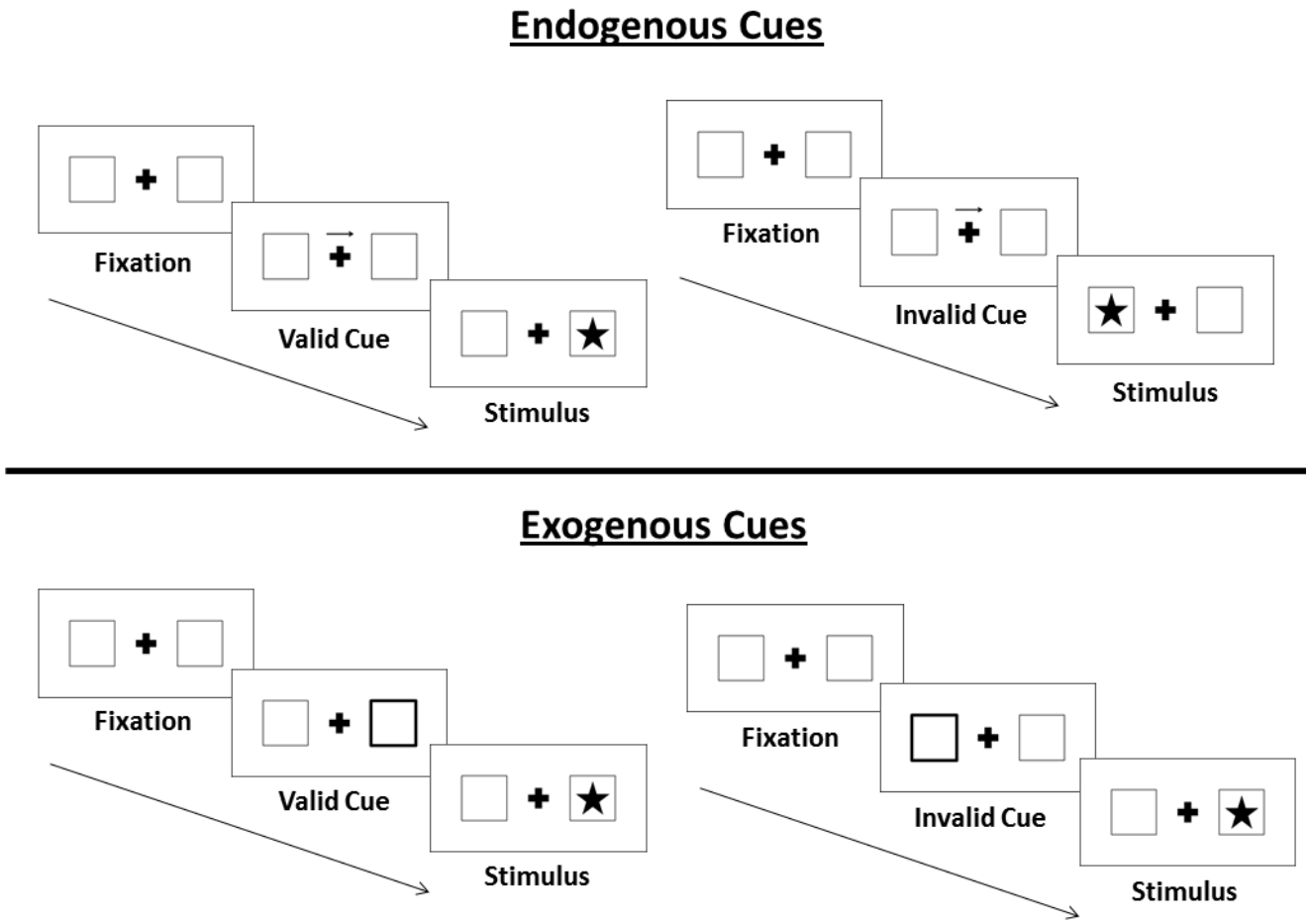
Does the stimulus contain an A or E?

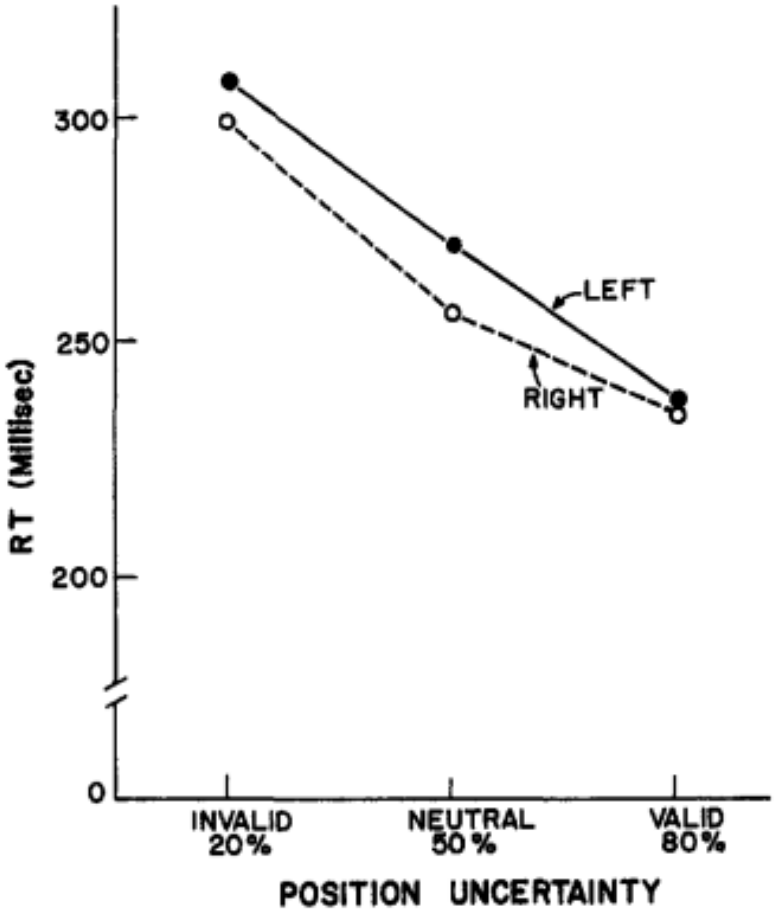
(press A or E)

SPATIAL ATTENTION TASK

POSNER CUING TASK

POSNER CUING TASK

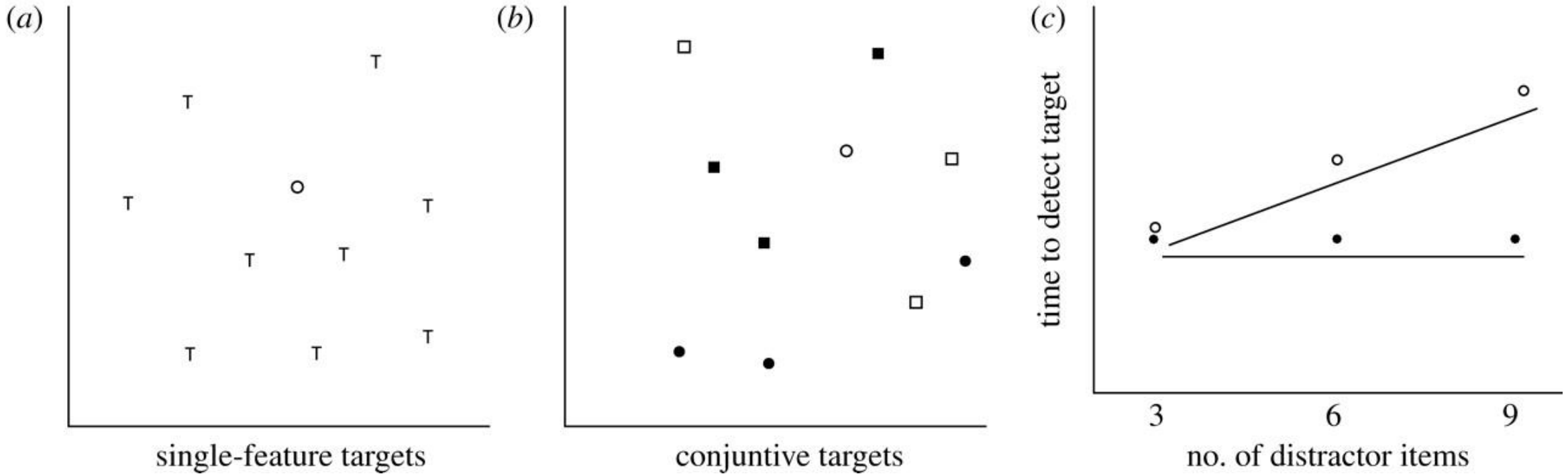




FEATURE SEARCH VS. CONJUNCTION SEARCH

TREISMAN

FEATURE VS. CONJUNCTION SEARCH



STROOP TASK

MULTIPLICITY OF MENTAL REPRESENTATIONS

MULTIPLICITY OF MENTAL REPRESENTATIONS

빨강

초록

파랑

노랑

검정

MULTIPLICITY OF MENTAL REPRESENTATIONS

빨강	XXXX
초록	XXXX
파랑	XXXX
노랑	XXXX
검정	XXXX

MULTIPLICITY OF MENTAL REPRESENTATIONS

빨강

초록

파랑

노랑

검정

XXXX

XXXX

XXXX

XXXX

XXXX

노랑

파랑

빨강

검정

초록

MODULES AND FUNCTIONS

SCRIPT

```
clear all
clc

% Set constants
applications = 30;
max_admits_allowed = 10;

IQmean = 110;
IQsd = 20;
SATQmean = 500;
SATQsd = 100;
SATVmean = 500;
SATVsd = 100;
ECsd = 10;
GPAmean = 2.0;
GPAsd = 10;
```

```
% Preallocate arrays using deal
[IQ SATQ SATV GPA Acad EC Dist] =
deal(zeros(applications,1));

% Generate dummy scores to test the program
IQ = IQmean + (randn(applications,1)) * IQsd;
SATQ = SATQmean + (randn(applications,1)) * SATQsd;
SATV = SATVmean + (randn(applications,1)) * SATVsd;
```

SCRIPT

```
GPA = GPAmean + (randn(applications,1)) * GPAsd;
EC = abs(randn(applications,1) * ECsd);
Dist = abs(randn(applications,1));
Acad = SATQ + SATV + 100 * GPA;

% Normalize the scores
Acad = (Acad - min(Acad)) ./ (max(Acad)-min(Acad));
EC = (EC -min(EC)) ./(max(EC)-min(EC));
Dist = (Dist - min(Dist)) ./(max(Dist)-min(Dist));

% Create a Scores matrix, including, in the final column,
% each student's total score
Scores = [[1:applications]' Acad EC Dist];
Scores(:,5) = [Acad + EC + Dist];
```

SCRIPT

```
 % Admit the top max_admits_allowed students (plus any ties)
SortedScores = sortrows(Scores,-5);
criterion = SortedScores(max_admits_allowed,5);
SortedScores(:,6) = 0;
SortedScores((SortedScores(:,5) >= criterion),6) = 1;
ScoresAndAcceptances = sortrows(SortedScores,1);

% Display the results
fprintf('App.\tAcad.\tExtra.\tDist.\tTotal\tAccept\n\n')
fprintf(...
  '%4d\t%6.2f\t%6.2f\t%6.2f\t%6.2f\t%4d\n',ScoresAndAcceptances')
fprintf('\r')
Students_Accepted = find(ScoresAndAcceptances(:,6));
fprintf('Accepted Students:\n');
fprintf('%3d',Students_Accepted);
fprintf('\n\n')
fprintf('Cutoff score: %5.03f\n', criterion);
```

DIVIDING INTO SMALL MODULES

```
%College_Admissions_Main.m

Clear_Start;
Set_Constants;
Generate_Dummy_Scores;
Normalize_Scores;
Create_Scores_Matrix;
Select_Students;
Display_Results;
```

SCRIPT VS. FUNCTION

FUNCTION

```
% function mymean.m
function myresult = mymean(inputarray);
myresult = sum(inputarray)/length(inputarray)
Return

meanD = mymean([1 3 5 7 9])
meanE = mymean([pi 1492 6.02])
meanF = mymean([1:10])

meanD =
     5


meanE =
  500.3872


meanF =
    5.5000
```

FUNCTION

```
% normalize.m
function y = normalize(x)
   y = (x-min(x))./(max(x)-min(x));
end
```

FUNCTION WITH MORE THAN ONE ARGUMENT

```
% normalize_split_two_args.m
% Splits array in first argument into
% lower and upper halves, using the
% criterion ('mean' or 'median')
% specified in the second argument

function [ly, uy] =
normalize_split_two_args(x,typeofsplit);

lx = [];
ux = [];

if strcmp(typeofsplit,'median')  % median split
    lx = x(x<=median(x));
    ux = x(x>median(x));

elseif strcmp(typeofsplit,'mean') % mean split
    lx = x(x<=mean(x));
    ux = x(x>mean(x));
else   % error feedback
    disp(['Error: An invalid type of split'...
    ' in the call to normalize_split_two_args']);
    [ly, uy] = deal(NaN);
    return
end

ly = (lx - min(lx)) ./ (max(lx)- min(lx));
uy = (ux - min(ux)) ./ (max(ux)- min(ux));
return
```

CREATING MULTIPLE FUNCTIONS IN A FILE

```
% mean_and_trimmed_mean.m
function [y,ty] = mean_and_trimmed_mean(x)
y = mean(x);
ty = mean(trimmed(x));
return

function zz = trimmed(w)
w = sort(w);
zz = [w(2:end-1)];
return
```

CREATING MULTIPLE FUNCTIONS IN A FILE

```
% ComputeMeans_Fails.m
x = randperm(5).^2;
[theMean theTrimmedMean] = mean_and_trimmed_mean(x)

function [y,ty] = mean_and_trimmed_mean(x)
y = mean(x);
ty = mean(trimmed(x));
return

function zz = trimmed(w)
w = sort(w);
zz = [w(2:end-1)];
return
```

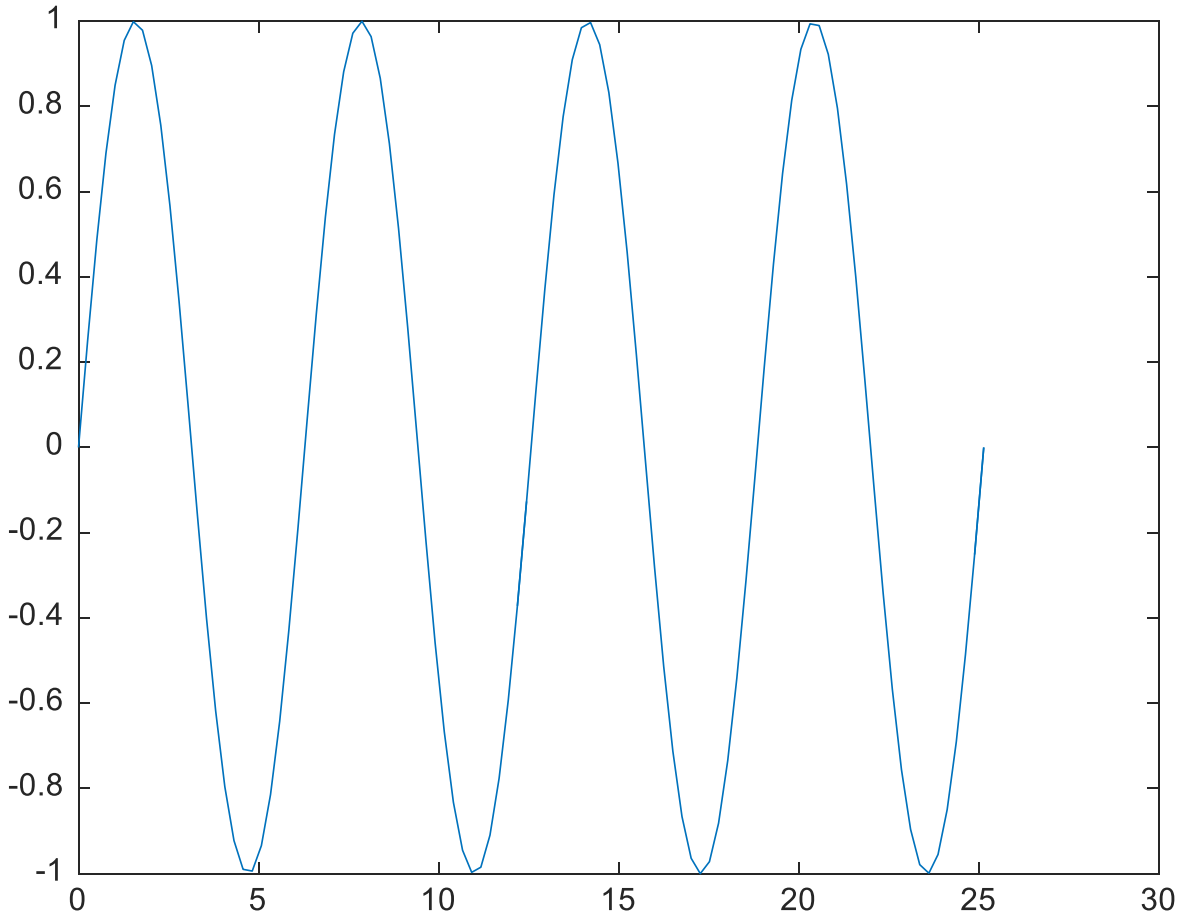
CREATING MULTIPLE FUNCTIONS IN A FILE

```
% ComputeMeans_Succeeds.m
function main
x = randperm(5).^2
[theMean theTrimmedMean] = mean_and_trimmed_mean(x)
return
end  % main function

function [y,ty] = mean_and_trimmed_mean(x)
y = mean(x);
ty = mean(trimmed(x));
return
end % mean_and_trimmed_mean

function zz = trimmed(w)
w = sort(w);
zz = [w(2:end-1)];
return
end % trimmed
```

PLOTS

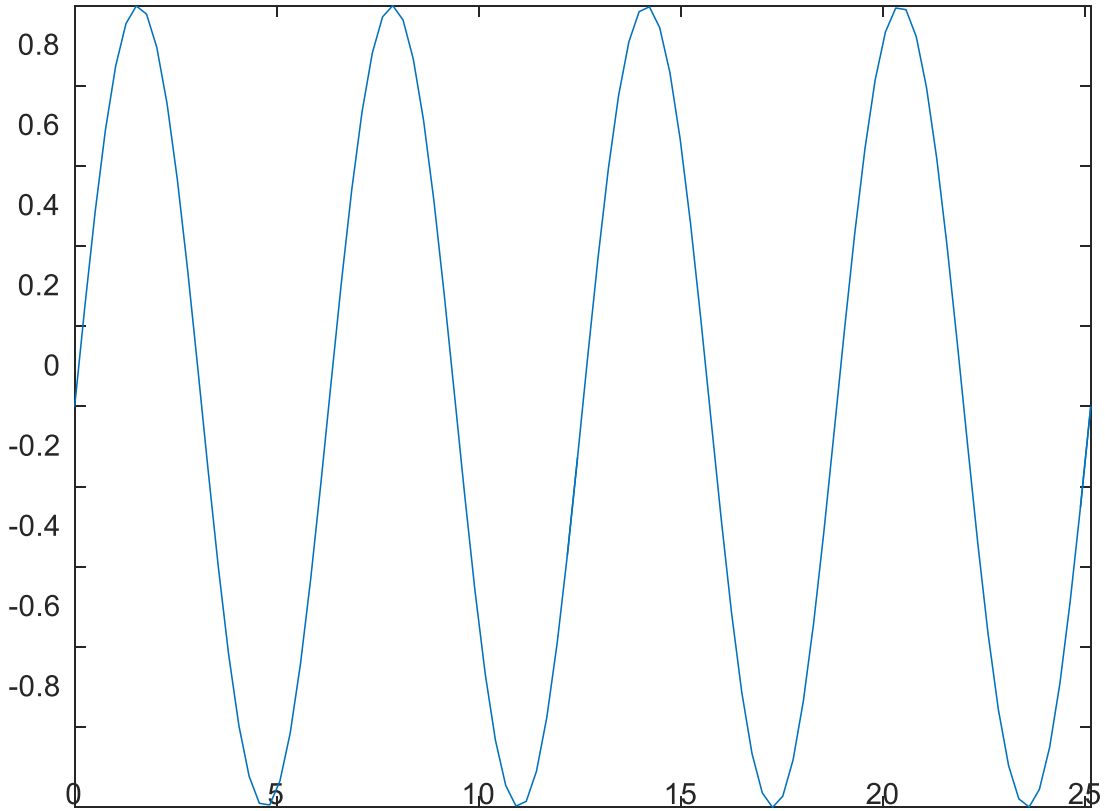
CONTROLLING PLOTS

```
clear all
close all
figure(1)
theta_rad = linspace(0,4*(2*pi),100);
plot(theta_rad,sin(theta_rad));
```



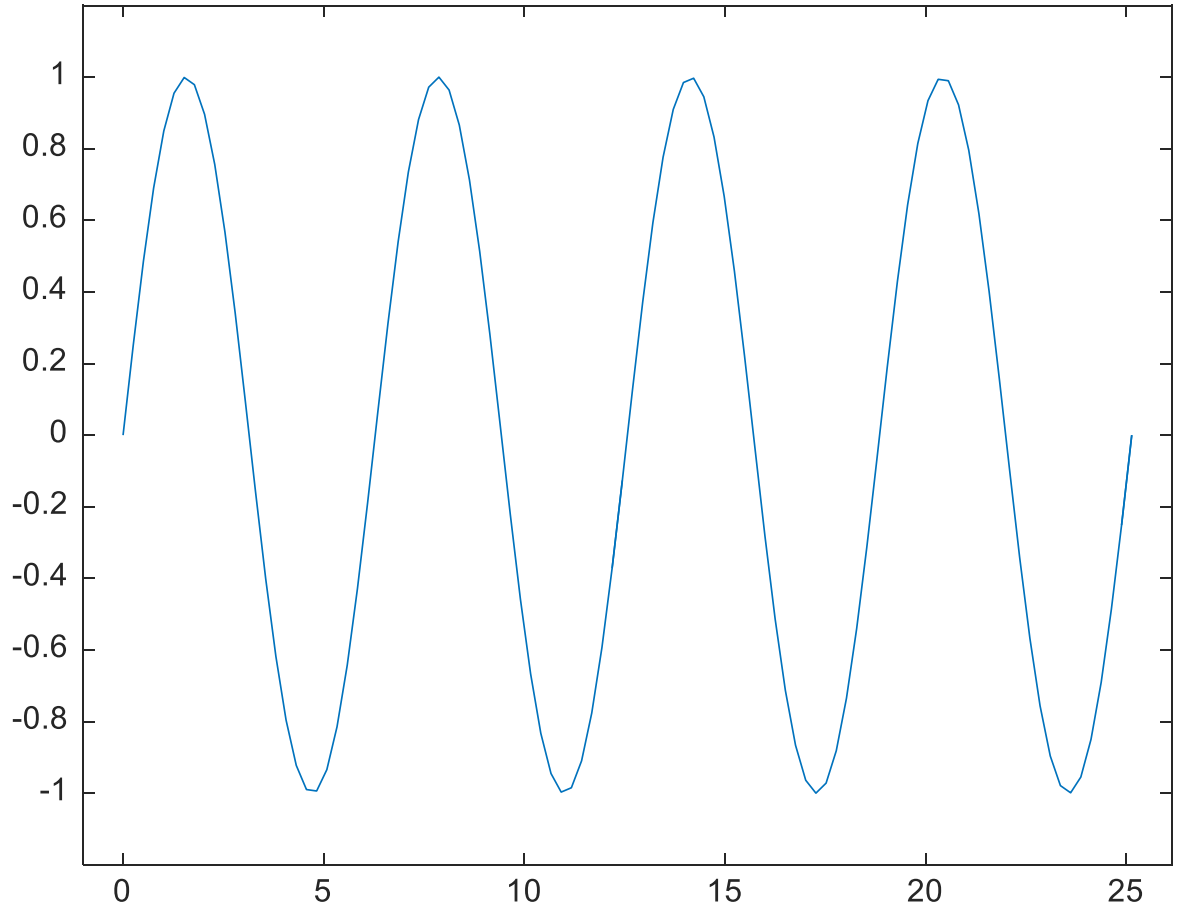
CONTROLLING PLOTS

```
clear all
close all
figure(1)
theta_rad = linspace(0,4*(2*pi),100);
plot(theta_rad,sin(theta_rad));
axis([min(theta_rad) max(theta_rad)
min(sin(theta_rad)) max(sin(theta_rad))]);
```



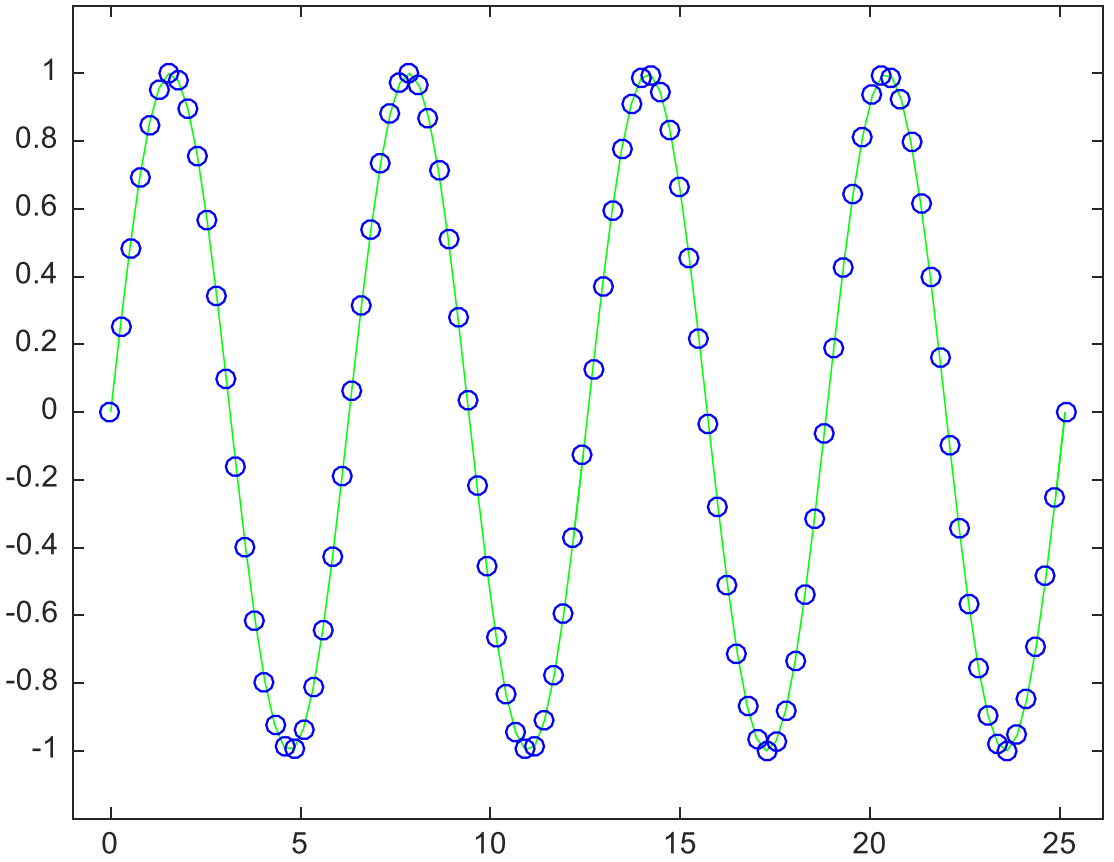
CONTROLLING PLOTS

```
clear all
close all
figure(1)
theta_rad = linspace(0,4*(2*pi),100);
x = theta_rad;
y = sin(theta_rad);
plot(x,y);
x_offset = 1;
y_offset = 0.2;
xlim([min(x)-x_offset, max(x+x_offset)]);
ylim([min(y)-y_offset, max(y+y_offset)]);
```



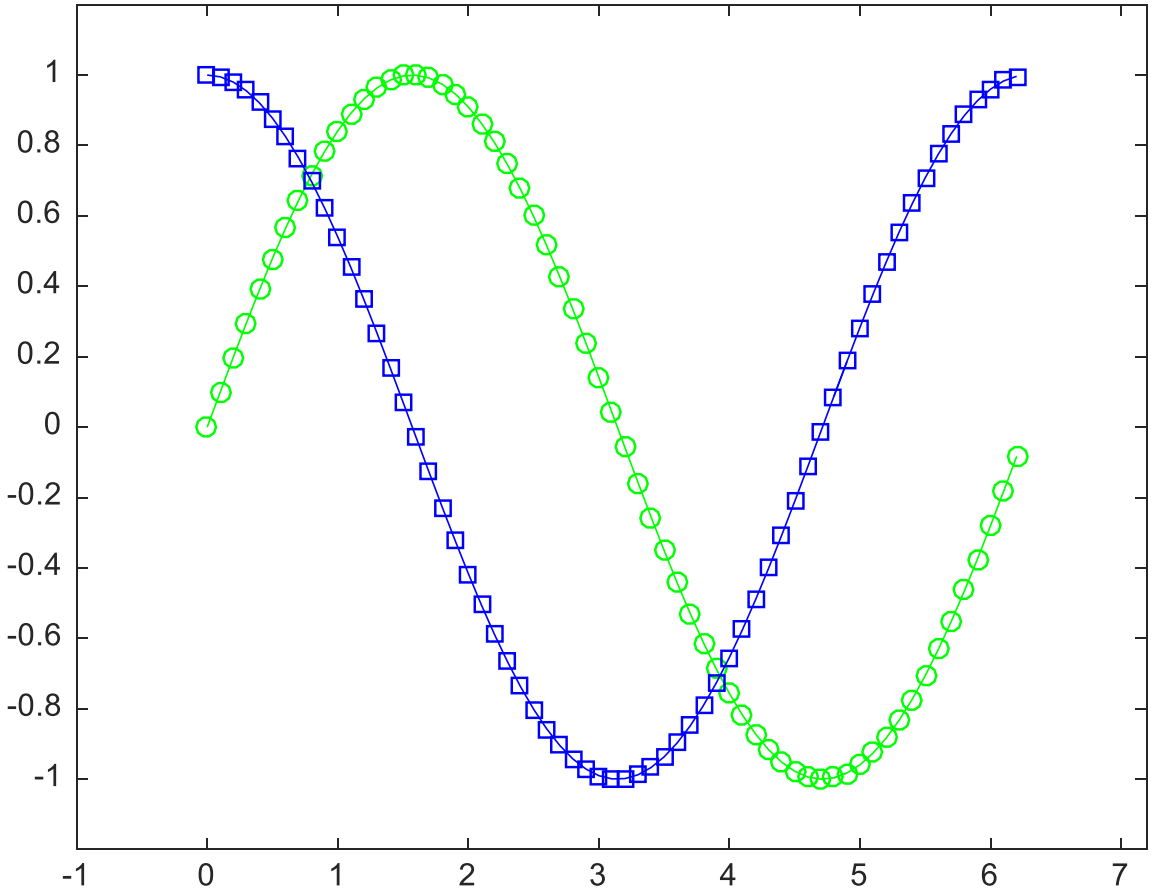
CONTROLLING PLOTS

```
clear all
close all
figure(1)
theta_rad = linspace(0,4*(2*pi),100);
x = theta_rad;
y = sin(theta_rad);
plot(x,y, 'g-');
hold on
plot(x,y, 'bo');
x_offset = 1;
y_offset = 0.2;
xlim([min(x)-x_offset, max(x+x_offset)]);
ylim([min(y)-y_offset, max(y+y_offset)]);
```



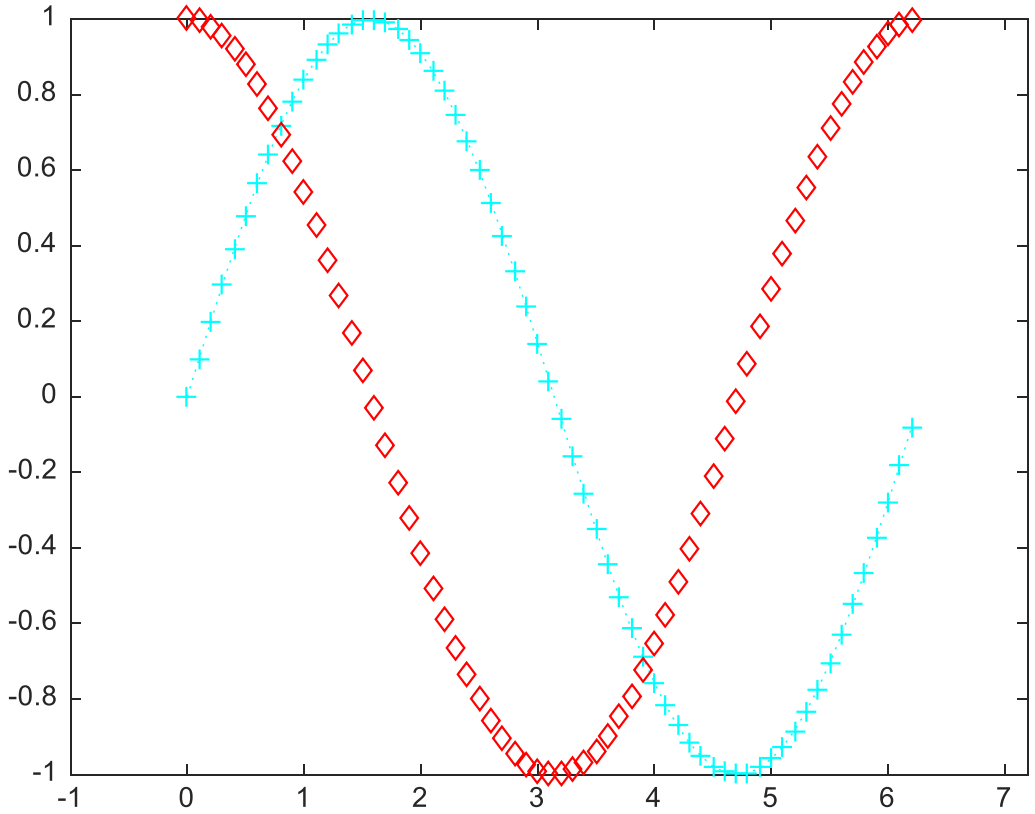
CONTROLLING PLOTS

```
clear all
close all
figure(1)
theta_rad = 0:.1:2*pi;
x = theta_rad;
y = sin(theta_rad);
plot(x,y,'go-');
hold on
y = cos(theta_rad);
plot(x,y,'b-s');
x_offset = 1;
y_offset = 0.2;
xlim([min(x)-x_offset, max(x+x_offset)]);
ylim([min(y)-y_offset, max(y+y_offset)]);
shg
```



CONTROLLING PLOTS

```
figure(1)
theta_rad = 0:.1:2*pi;
x = theta_rad;
plot(x,sin(theta_rad),'c+:', x, cos(theta_rad),
'rd');
```



CONTROLLING PLOTS

```
clear all
close all
figure(1)
theta_rad = 0:.1:8*pi;
x = theta_rad;
y = sin(x);
plot(x,y,'g-');
hold on
x_offset = 0;
y_offset = .2;
axis([min(x)-x_offset, max(x)+x_offset, ...
      min(y)-y_offset, max(y+y_offset)]);
plot(x,y,'o','color','r','markersize',6,...

'markeredgecolor','k','markerfacecolor','r');
xlabel('Time');
ylabel('Happiness');
title('Life has its ups and downs');
```