Matlab version of typical Mathcad calculations

Basic calculations - solution to quadratic equation:

$$ax^2 + bx + c = 0$$

```
a=1; b=2; c=3;
```

$$x_{1st} = \frac{-b + \sqrt{b^2 - 4ac}}{2a} \qquad x_{2nd} = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

```
x_1st = (-b + sqrt(b^2 - 4*a*c)) / (2*a);
x_2nd = (-b - sqrt(b^2 - 4*a*c)) / (2*a);
disp (['x_1st = ' num2str(x_1st) '    x_2nd =' num2str(x_2nd)]);
```

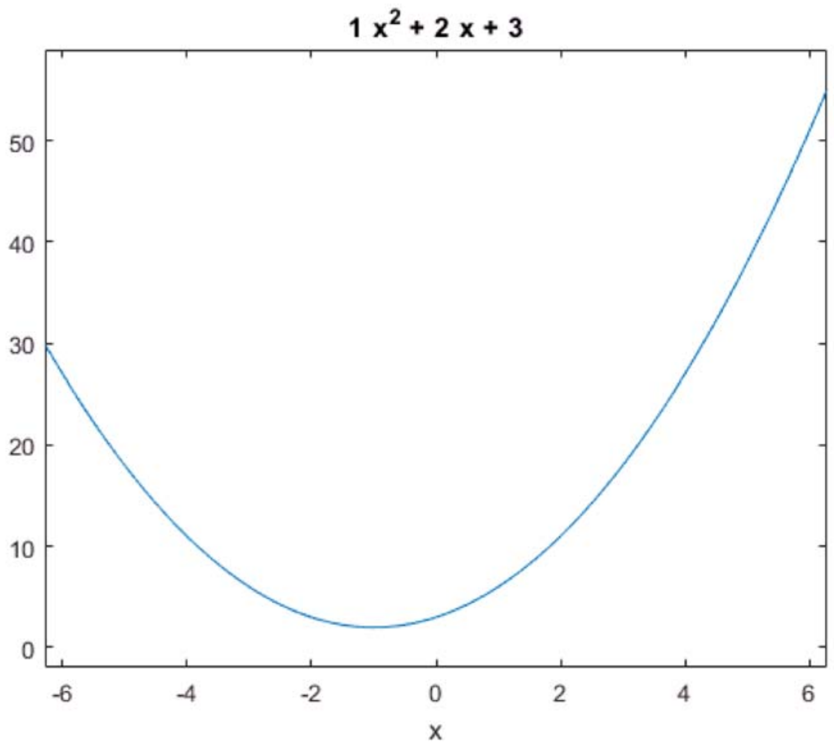
```
x_1st = -1+1.4142i    x_2nd =-1-1.4142i
```

```
% checking results in my_quadratic function:

% function [f] = my_quadratic(x, a, b, c)
% % Function to evaluate the quadratic function with predefined a, b, c
% f = a*x.^2 + b*x + c;
% end

fprintf ('f(x_1st) = %i     f(x_2nd) = %i\n', my_quadratic(x_1st, a, b, c), my_quadratic(x_2nd, a, b, c));
```

```
f(x_1st) = -4.440892e-16     f(x_2nd) = -4.440892e-16
```

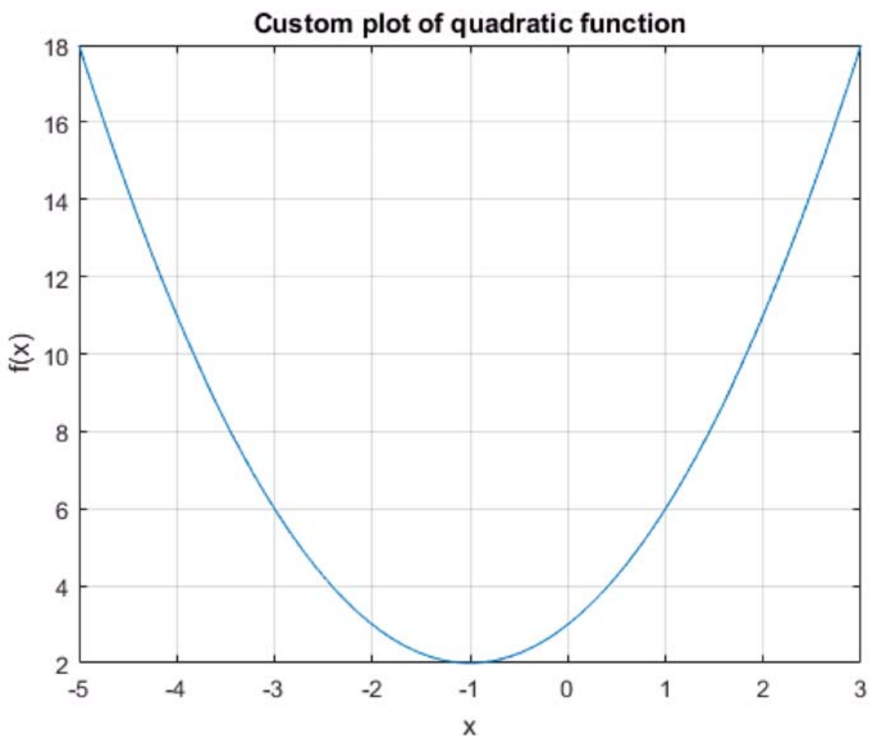
Plotting a function with automated ranges and number of points

```
ezplot('1*x^2 + 2*x + 3')
```



Plotting a function using a vector of values, with custom display

```
x = -5 : 0.05 : 3;
y = my_quadratic(x, a, b, c);
plot (x, y)
title ('Custom plot of quadratic function');
xlabel('x');
ylabel('f(x)');
grid on
```



```
% Display both ends of x vector
x_length = length(x);
for i = 1:15
    x_lower(i) = x(i);
    x_upper(i) = x(x_length - 15 + i);
end
x_lower
```

```
x_lower =
   -5.0000   -4.9500   -4.9000   -4.8500   -4.8000   -4.7500   -4.7000   -4.6500   -4.6000   -4.5500   -4.5000
```

```
x_upper
```

```
x_upper =
    2.3000    2.3500    2.4000    2.4500    2.5000    2.5500    2.6000    2.6500    2.7000    2.7500    2.8000
```

Using units and formatted display

(unit conversion functions available in Aerospace Toolbox only)

```
% m = convmass (100, 'lbm', 'kg');
m = 100 / 2.204622622;    % conver lbm to kg
% v = convvel (60, 'mph', 'm/s');
v = 60 * 0.44704;     % convert mph to mps
% a = convacc (20, 'ft/s^2', 'm/s^2');
a = 20 * 0.3048;     % convert fps2 to mps2
p = m*v
```

```
p = 1.2166e+03
```

```
F = m*a;
% convforce(F, 'N', 'lbf')
F = F / 4.448       % conver N to lbf
```

```
F = 62.1650
```

Symbolic algebra

$$\frac{x}{(2x - 3xy)} = \frac{(x-2)^2}{y+2}$$

solve equation:

```
syms x y
eqn = x / (2*x - 3*x*y) == (x-2)^2/(y+2);
x_ans = solve (eqn)
```

```
x_ans =
```

$$\begin{pmatrix} \frac{6\,y + \sqrt{-(3\,y-2)\;(y+2)} - 4}{3\,y-2} \\ -\frac{\sqrt{-(3\,y-2)\;(y+2)} - 6\,y + 4}{3\,y-2} \end{pmatrix}$$

evaluate solutions:

```
subs(x_ans, 'y', 5)
```

```
ans =
```

$$\begin{pmatrix} 2 + \frac{\sqrt{91}\,\mathrm{i}}{13} \\ 2 - \frac{\sqrt{91}\,\mathrm{i}}{13} \end{pmatrix}$$

Symbolic calculus

```
syms x a
fx = (x - a)^2 + 10*sin(2*x)/x
```

```
fx =
```

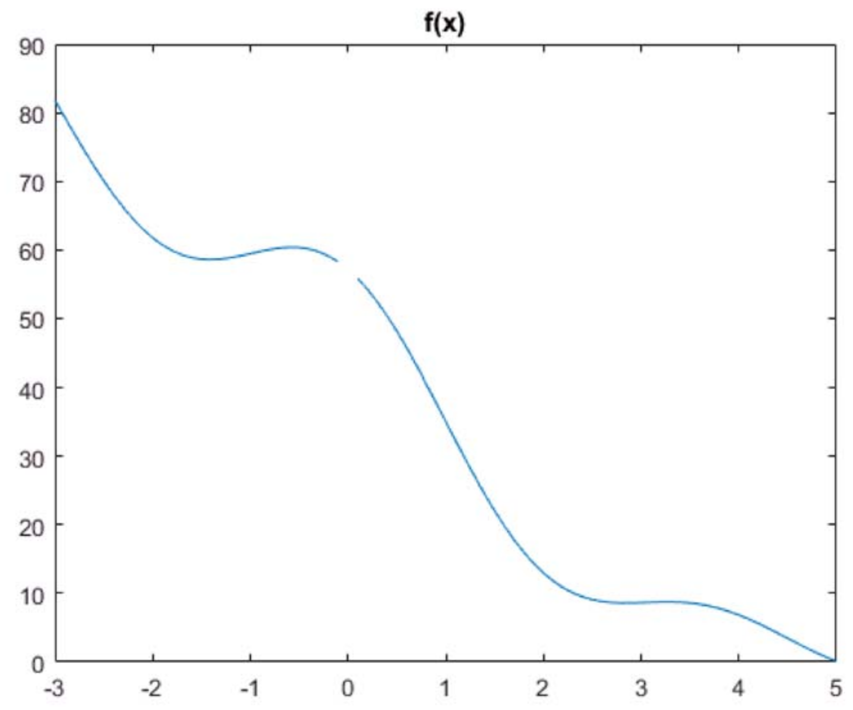
$$\frac{10\sin(2\,x)}{x} + (a-x)^2$$

```
dfx = diff (fx)
```

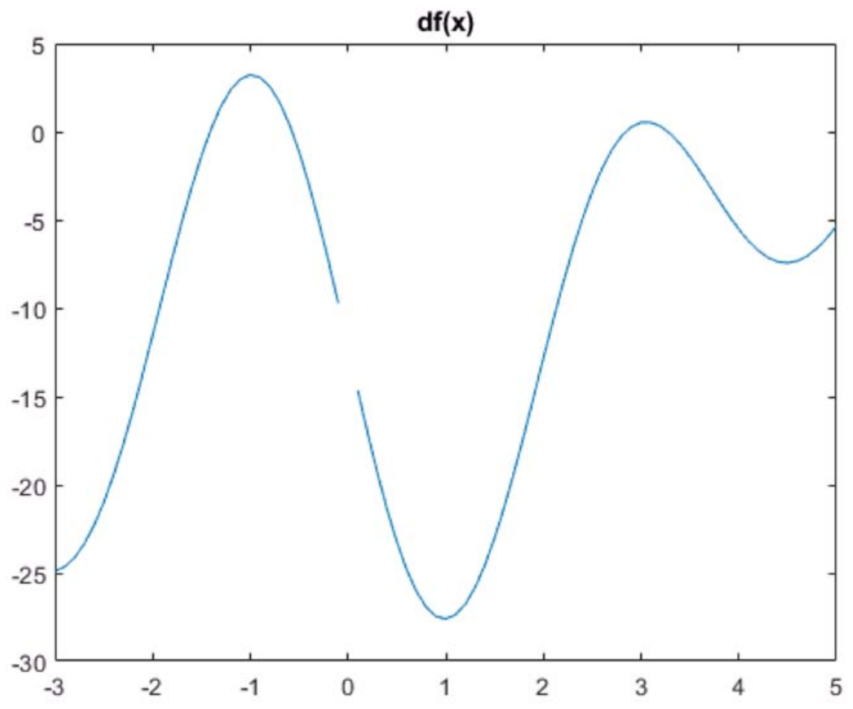
```
dfx =
```

$$2\,x - 2\,a + \frac{20\cos(2\,x)}{x} - \frac{10\sin(2\,x)}{x^2}$$

```
x = -3:0.1:5;
a = 6.096;
y = eval(fx);
dy = eval(dfx);
plot (x, y)
title ('f(x)')
```



```
plot (x, dy)
title ('df(x)')
```



Vector and matrix calculations

$$v = \begin{bmatrix} -1 \\ -2 \end{bmatrix}$$

```
vx = -1; vy = -2;
v = [vx; vy];
v = vx + j*vy
```

```
v =
  -1.0000 - 2.0000i
```

$$|v|$$

```
abs(v)
```

```
ans = 2.2361
```

$$v \cdot v$$

```
dot(v,v)
```

```
ans = 5
```

```
v_ang = angle(v)*180/pi;
display (['angle of v = ' num2str(v_ang) ' deg']);
```

```
angle of v = -116.5651 deg
```

```
display (['polar form of v = ' num2str(v_mag) ' < ' num2str(v_ang)]);
```

```
polar form of v = 2.2361 < -116.5651
```

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 1 & 5 \\ 0 & -2 & 3 \end{bmatrix}$$

```


A = [1 2 3; 2 1 5; 0 -2 3];
```

$$A^{-1}$$

```
inv(A)
```

```
ans =
   -1.1818    1.0909   -0.6364
    0.5455   -0.2727   -0.0909
    0.3636   -0.1818    0.2727
```

$$I = AA^{-1}$$

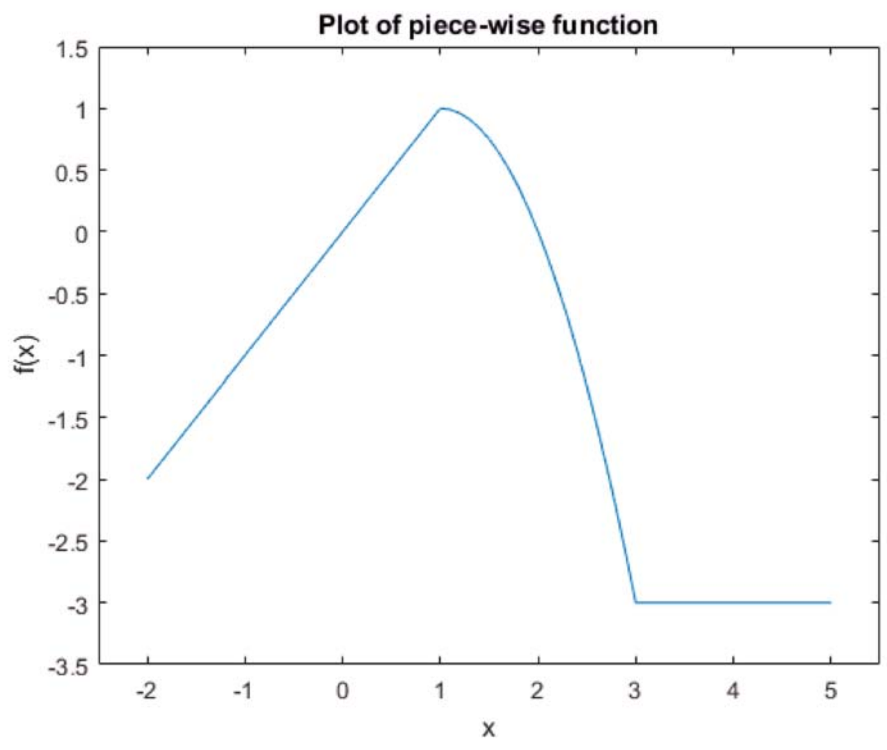
```
I = A * A_inv
```

```
I =
    1.0000   -0.0000   -0.0000
   -0.0000    1.0000   -0.0000
    0.0000   -0.0000    1.0000
```

Programming a piecewise function

$$f(x) = \begin{cases} x & x < 1 \\ -(x-1)^2 + 1 & 1 \le x \le 3 \\ -3 & x > 3 \end{cases}$$

```
% function [f] = my_piece_wise(x, a, b, c)
% % Function to evaluate the quadratic function with predefined a, b, c
% if (x < 1)
%     f = x;
% elseif ((x >= 1) && (x <= 3))
%     f = -(x-1)^2 + 1
% else
%     f = -3
% end

x = -2 : 0.1 : 5;
clear y;
for (i = 1 : length(x))
    y(i) = my_piece_wise(x(i));
end
plot (x, y);
title ('Plot of piece-wise function');
xlabel('x');
ylabel('f(x)');
axis([-2.5 5.5 -3.5 1.5]);
```



General programming problem example

Find the sum of the first N numbers divisible by 3

```
% function [i total] = my_program(N)
% % Function to calculate the sum of the first N numbers divisible by 3
% i = 0;
% n = 0;
% total = 0;
%
% while (n < N)
%     i = i + 1;
%     remainder = mod (i, 3);
%     if (remainder == 0)
%         total = total + i;
%         n = n + 1;
%     end
% end

N = 10000;
[i, total] = my_program(N)
```

```
i = 30000
total = 150015000
```

Finding roots

$$f(x) = 2x^2 - 4\sin(x) - 2$$

```
% function [f] = my_root_func(x)
% % Function to evaluate the quadratic function with predefined a, b, c
% f = 2*x^2 - 4*sin(x) - 2;
% end
```

roots for different guesses:

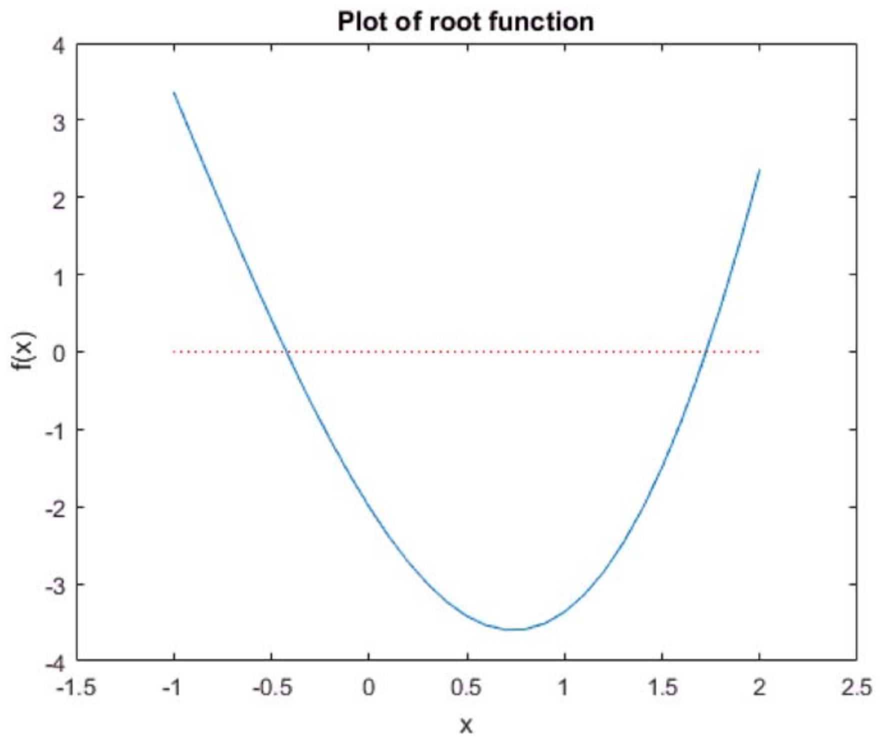
```
x0 = 1;
fzero (@my_root_func, x0)
```

```
ans = 1.7252
```

```
x0 = -1;
fzero (@my_root_func, x0)
```

```
ans = -0.4230
```

```
x = -1 : 0.1 : 2;
y = 2*x.^2 - 4*sin(x) - 2;
plot (x, y);
title ('Plot of root function');
xlabel('x');
ylabel('f(x)');
hold on;
x = [-1 2];
y = [0 0];
plot (x,y,'LineStyle',':','Color',[1 0 0]);
axis([-1.5 2.5 -4 4]);
hold off;
```



Solving a set of nonlinear equations

$$x = 2 - y^2$$

$$y = \frac{\sin(x)}{x} + \mathrm{xy}$$

```
% function [ F ] = my_non_linear_equations( x )
% % Define set of nonlinear equations to be solved numerically
% F = [x(1) - 2 + x(2)^2;  x(2) - sin(x(1))/x(1) + x(1)*x(2)];
% end

x0 = [1; 1];  % initial guesses
[x_sol,fval] = fsolve(@my_non_linear_equations,x0);
```

```
Equation solved.

fsolve completed because the vector of function values is near zero
as measured by the default value of the function tolerance, and
the problem appears regular as measured by the gradient.

<stopping criteria details>
```

```
x_sol
```

```
x_sol =
    0.2517
    1.3222
```

```

% Checking results (solving symbolically and plotting)
syms x y;
fa = solve(x == 2 - y^2, y)
```

fa =

$$\begin{pmatrix} \sqrt{2-x} \\ -\sqrt{2-x} \end{pmatrix}$$

```
fb = solve(y == sin(x)/x + x*y, y)
```

fb =

$$\frac{\sin(x)}{x - x^2}$$

```
x = x_sol(1)
```

```
x = 0.2517
```

```
eval(fa(1))
```

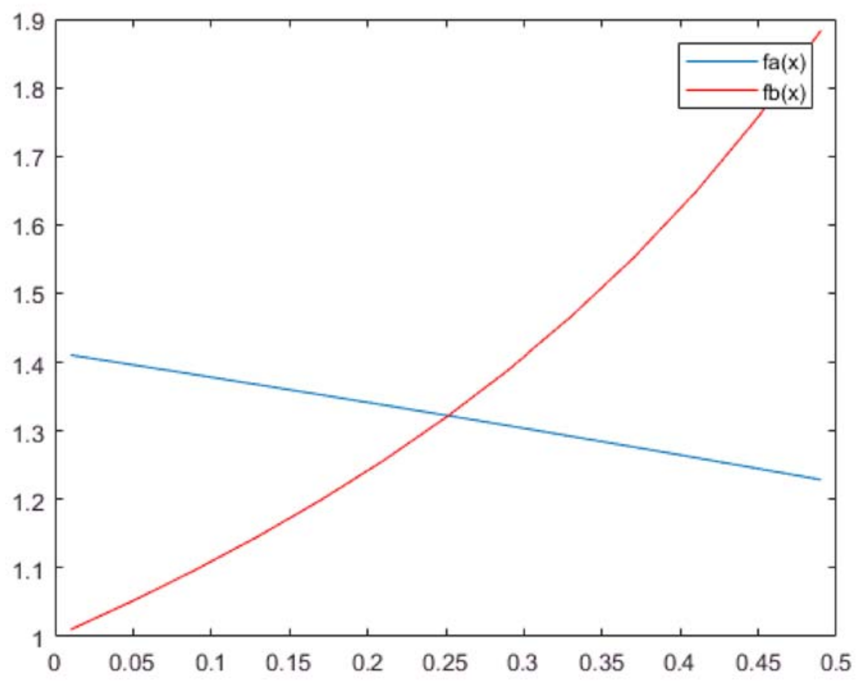
```
ans = 1.3222
```

```
eval(fb)
```

```
ans = 1.3222
```

```
x = 0.01 : 0.04 : 0.5;
ya = eval (fa(1));
yb = eval (fb);
plot (x, ya);
hold on;
plot (x, yb, 'Color',[1 0 0]);
legend ('fa(x)', 'fb(x)');
```

```
hold off;
```



Iterative calculations

```
clear x y;
x(1) = 1, y(1) = 1
```

```
x = 1
y = 1
```

```
for i = 1 : 6
    x(i+1) = x(i) + 2;
    y(i+1) = (x(i) + x(i+1)) / 2;
end
x
```

```
x =
     1     3     5     7     9    11    13
```

```
y
```

```
y =
     1     2     4     6     8    10    12
```

Finding an optimal solution given constraints

(requires Optimization Toolbox)

maximize

$$z(x, y) = (x - 1)^2 - x\sin(y)$$

subject to

$$x > -2$$

$$x < 2y^2 + 3$$

$$-3 < y < 5$$

```
% function [ F ] = my_objfun( x )
% % Function definition for constrained optimization problem
% % (minus sign in front for max vs. min)
% F = - ((x(1)-1)^2 - x(1)*sin(x(2)));
% end

% function [c, ceq] = my_confun(x)
% % Nonlinear inequality constraints
% c = [-x(1) - 2; x(1) - 2*x(2)^2 - 3; x(2) - 5; -x(2) - 3];
% % Nonlinear equality constraints
% ceq = [];

x0 = [1; 1];
[x,fval] = fmincon(@my_objfun,x0,[],[],[],[],[],[],@my_confun);
```

```
Local minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the default value of the optimality tolerance,
and constraints are satisfied to within the default value of the constraint tolerance.

<stopping criteria details>
```

```
x
```

```
x =
   -2.0000
    1.5708
```

```
-fval   % minus for max vs. min
```

```
ans = 11.0000
```