# Nonlinear Equations in Matlab

Jake Blanchard
University of Wisconsin - Madison

# Introduction

- Nonlinear algebraic equations are encountered in many scientific applications
- **fzero** will solve single equations
- Matlab's **fsolve** command can solve these
- Nonlinearity implies potential for
  - No solution
  - Multiple solutions
- You may need a pretty good guess at solution

# Model Problem

$$x^2 + 2y^2 - 5x + 7y = 40$$

$$3x^2 - y^2 + 4x + 2y = 28$$

# Convert to Functions

$$x^2 + 2y^2 - 5x + 7y - 40 = 0$$

$$3x^2 - y^2 + 4x + 2y - 28 = 0$$

# How Does **fsolve** work?

- This command finds the roots of systems of functions
- We supply a set of functions and Matlab will find all the independent variables such that all the functions are zero (or near-zero)
- Solution is iterative, so we must provide guess

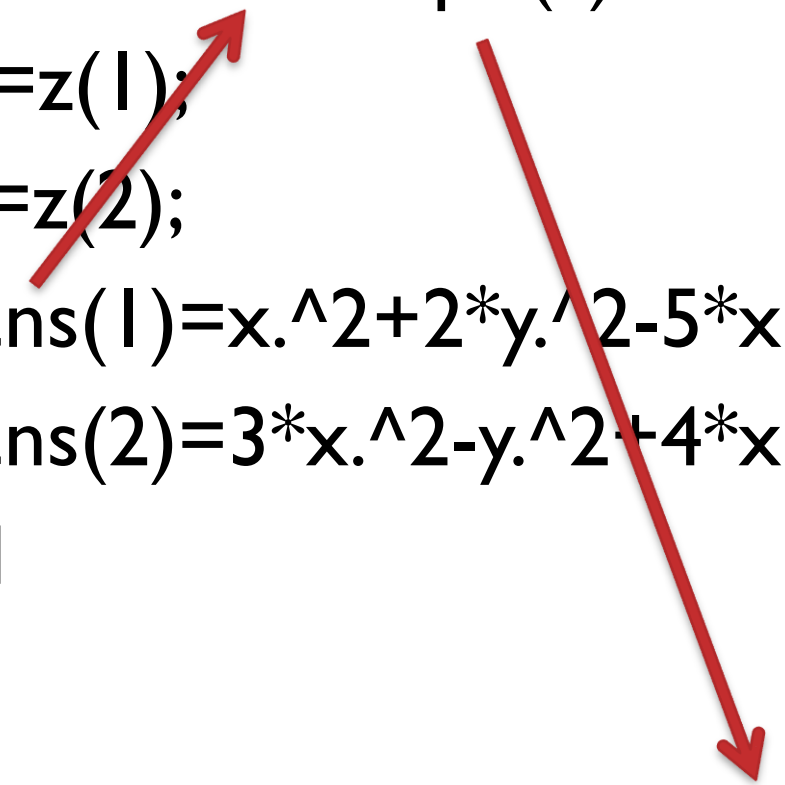
# Define Functions

```
function fcns=eqns(z)
   x=z(1);
   y=z(2);
   fcns(1)=x.^2+2*y.^2-5*x+7*y-40;
   fcns(2)=3*x.^2-y.^2+4*x+2*y-28;
end
```

**Save this to a file called eqns.m**

# Define Functions

```
function fcns=eqns(z)
   x=z(1);
   y=z(2);
   fcns(1)=x.^2+2*y.^2-5*x+7*y-40;
   fcns(2)=3*x.^2-y.^2+4*x+2*y-28;
end
```



**Save this to a file called eqns.m**

# Calling the solver

```
guess=[2 3];
result=fsolve(@eqns, guess)
```

Or

```
guess=[2 3];
[result, fval, exitflag, output]
                        =fsolve(@eqns, guess)
```

# The Full Code

```
function solveeqs()
guess=[2 3];
[result, fval, exit, output]=fsolve(@eqns, guess);
result
fval
eqns(guess)
output
end

function fcns=eqns(z)
x=z(1);
y=z(2);
fcns(1)=x.^2+2*y.^2-5*x+7*y-40;
fcns(2)=3*x.^2-y.^2+4*x+2*y-28;
end
```