# Implementation of implicit integrators in Matlab/Octave

Marc R. Roussel

November 21, 2019

# Passing parameters to functions

- Matlab functions take formal arguments, but sometimes we need to pass a parameter to a function in a way that is invisible, perhaps because the function is to be used as a function argument itself.

Method 1: Declare the parameter to be `global`.

```
global k m
```

Put this line both in the main script before assigning values to the parameters, and in the function.

Method 2: Anonymous functions

```
@(x)f(x,k)
```

This creates a new function of `x` from a function that actually depends on both `x` and `k`.

# Iterative solution method

- Equations to solve for $(x_{j+1}, p_{j+1})$:

$$\frac{x_{j+1} - x_j}{h} = \frac{H(x_j, p_{j+1}) - H(x_j, p_j)}{p_{j+1} - p_j}$$
$$\frac{p_{j+1} - p_j}{h} = -\frac{H(x_{j+1}, p_{j+1}) - H(x_j, p_{j+1})}{x_{j+1} - x_j}$$

- Rewrite in the Euler-like form

$$x_{j+1} = x_j + h\frac{H(x_j, p_{j+1}) - H(x_j, p_j)}{p_{j+1} - p_j}$$
$$p_{j+1} = p_j - h\frac{H(x_{j+1}, p_{j+1}) - H(x_j, p_{j+1})}{x_{j+1} - x_j}$$

# Iterative solution method

$$x_{j+1} = x_j + h\frac{H(x_j, p_{j+1}) - H(x_j, p_j)}{p_{j+1} - p_j} \tag{1a}$$

$$p_{j+1} = p_j - h\frac{H(x_{j+1}, p_{j+1}) - H(x_j, p_{j+1})}{x_{j+1} - x_j} \tag{1b}$$

- Problem: $x_{j+1}$ and $p_{j+1}$ both appear in the right-hand sides of these equations.
- Generate a guess for $x_{j+1}$ and $p_{j+1}$ using an explicit method, then substitute this guess into equations (1) to refine this approximate solution.
  Iterate until the variables change negligibly from one iteration to the next.

# Using `fsolve()`

- Instead of solving the equations yourself by iteration, you can use the Matlab/Octave function `fsolve()`.
- Warning: In Matlab, `fsolve()` is part of the Optimization Toolbox. It may not be available in a "vanilla" Matlab installation.
- Syntax:

```
v = fsolve(f,v0,options)
```

  Assumption: you want to solve the equation `f(v) = 0`, where `v` is a vector containing your variables.
  `v0` is an initial guess vector.
  On output, `v` is the solution.

# `fsolve` options

- The options used by `fsolve` have different names in Matlab and Octave.

`FunctionTolerance` (Matlab)/`TolFun` (Octave) is the maximum value of `f` required for the algorithm to stop.

`StepTolerance` (Matlab)/`TolX` (Octave) is the maximum difference between one iterate and the next for the algorithm to stop.

- An option structure is created by `optimset()`.
  ```
  opts = optimset('TolX',1e-8,'TolFun',1e-4);
  ```
- An option structure (e.g. `opts` above) is passed as the final argument of `fsolve()`.