HW #4 Sample code

Trying to use symbolic functions for complicated derivatives in MATLAB is pretty rough, because it'll probably leave you with more lines of code for substituting symbolic functions and variables in and out for one another than lines that get used to solve the actual problem. My preference is to just stick to normal symbolic variables. It's actually possible to do the entire assignment without creating a single symbolic function (e.g. `syms q(t)`)!

I've created the following example to show the general idea behind how to avoid using symbolic functions, and instead just use plain symbolic variables. The example will create a function, f, of two joint variables, q1 and q2. The goal is to differentiate f(q1,q2) with respect to t, treating q1 and q2 as functions of t without explicitly defining them that way. All of the text in fixed-width font can be run directly in MATLAB, too!

Define two symbolic joint variables normally. They're real-valued:

```
syms q1 q2 real
```

Suppose you have some function, f, of the joint variables that you want to differentiate:

```
f = q1^2 + cos(q2)
```

The goal is to find df/dt. Keep in mind, we want to treat q1 and q2 as functions of t, but they haven't been defined that way in MATLAB.

f is a function of q1 and q2, and q1 and q2 are both functions of t. So using the multivariate chain rule, df/dt is as follows (in math notation, not MATLAB code):

```
df/dt = df/dq1 * dq1/dt + df/dq2 * dq2/dt
```

To do that in MATLAB, we need symbolic variables (dq1 and dq2) representing joint velocities (dq1/dt and dq2/dt), since we can't just call diff(q1,t) or diff(q2,t) to get them.

They are real-valued:

```
syms dq1 dq2 real
```

So finding df/dt in MATLAB is just the following:

```
df_dt_method1 = diff(f,q1)*dq1 + diff(f,q2)*dq2
```

Alternatively (simpler for large numbers of joints - much more extensible!):

```
q_sym = [q1 q2]
dq_sym = [dq1 dq2]
df_dt_method2 = jacobian(f,q_sym)*dq_sym.'
```

Either way, you get the same result. Using this approach you can get from a symbolic Lagrangian to the full symbolic Euler-Lagrange dynamics in a grand total of four lines of MATLAB.

Then, to solve for df/dt for some given joint positions and velocities:

```
q_num = [1 2]
dq_num = [-0.2 1.2]
df_dt_numerical = eval(subs(df_dt_method2,[q_sym dq_sym],[q_num
dq_num]))
```