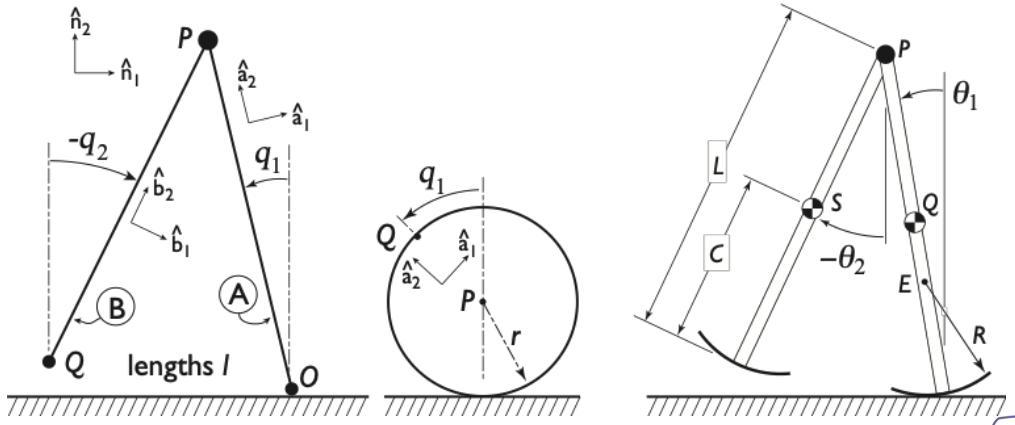
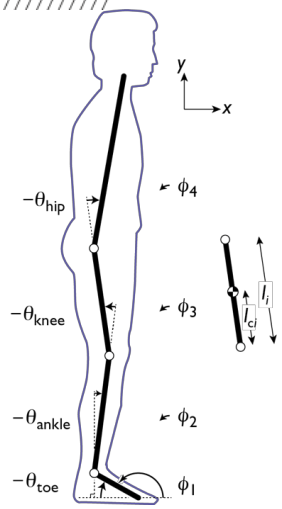
Homework 01

1. Use graphical kinematics to determine the velocities and accelerations of the points (P, Q, etc. as labeled) in the systems illustrated here. You can define unit vectors as necessary, and draw in copies of the unit vectors and the magnitudes of the associated vectors, as demonstrated in class. Solution should be in the form of a drawing, which is optional to turn in (via uploaded image).



2. Jacobian relationships may be formed for a variety of kinematic transformations. The system at right has configuration described by two types of angles: Joint angles θ_i and Segment angles ϕ_i. Here, joint angle θ_i ($i = 1, \ldots, 4$ for toe through hip) describes segment i's angle relative to neighboring segment $i-1$, with θ_i increasing positively in the joint extension direction. Segment angle ϕ_i describes segment i's angle, increasing in the counter-clockwise direction relative to vertical. Of course, a variety of conventions are employed, and so it is helpful to develop transformations between different conventions.



   The transformation between segment and joint angles looks something like $\theta_4 = \phi_4 - \phi_3$. There is a Jacobian relationship between these angles.

   Another Jacobian of interest is the one for the transformation from segment/joint angles to segment positions and orientations. A full description of translational position and angular orientation is called the "pose," such as

$$\mathbf{X} \triangleq \begin{bmatrix} x_{c1} \\ y_{c1} \\ \phi_{c1} \\ x_{c2} \\ y_{c2} \\ \phi_{c2} \\ \vdots \end{bmatrix}.$$

A. Find the segments-joints Jacobian that yields $\underline{\theta} = J_{\theta\phi}\ \underline{\phi}$, and its inverse, $J_{\phi\theta}$. Note that the transformation between angles includes a constant offset, but not the Jacobian. Why?
B. Find the pose Jacobian, $\dot{\mathbf{X}} = J_{X\phi}\dot{\underline{\phi}}$. (No need to write out entire matrix, but a few representative rows.)
C. Suppose the pose Jacobian is desired for joint angles. How can $J_{X\phi}$ and $J_{\theta\phi}$ yield $J_{X\theta}$?

3. Although Jacobians often refer to kinematic relationships, in general they may refer to the vector partial derivative of an arbitrary function. Write a Matlab function that calculates such a numerical Jacobian. Use a first-order finite difference approximation,

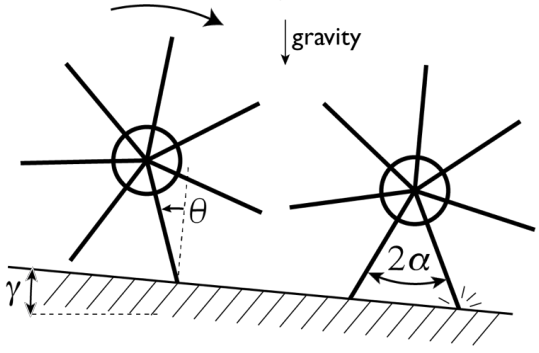
$$J(x_0) = \left.\frac{\partial f}{\partial x}\right|_{x_0} \approx \frac{f(x_0 + \Delta x) - f(x_0)}{\Delta x}.$$

As an example, run `testfgradient.m`, which tests function `fgradient.m`. You can use the latter as a starting point for your own `fjacobian.m`.
A. Add or modify code, and test your Jacobian function with `testfjacobian.m`.
B. The final test is actually a demonstration of the computational error. Notice that the error decreases with step size only to a point. In a sentence or two, explain why.

The code here is an example of test-driven development. The test scripts can be run on their own (and as individual cells), but Matlab can also organize and combine multiple such tests, e.g. `runtests('testfgradient')`. It is considered good practice to develop code along with unit tests together.

4. Write a function to numerically find the root of a vector function, using Newton's method. A test suite `testfindroot` and some starter code `findroot_startercode` are provided.
A. Use `findroot_startercode` to produce `findroot`, and make sure it passes tests.
B. Demonstrate that a poor enough initial guess can cause an unintended root to be found. In the test suite, the intended root of `fhandle1` is at +4, but there are two "unintended" roots.
C. Demonstrate that `fhandle2` in the test suite has a root at zero, but that `findroot` cannot converge to the proper solution.
D. Test your function's ability to handle vector-valued functions and roots, using `fhandle3`.

5. Simulate the stance phase of the rimless wheel model. The rimless wheel is a simple inverted pendulum, with equation of motion $\ddot{\theta} - \frac{1}{(1+r_{\text{gyr}}^2)}\sin(\theta - \gamma) = 0$, where the stance leg angle θ is measured counter-clockwise with respect to the ground normal (i.e., not exactly vertical), and the ground has a downward slope of γ. The simulation should start at $\theta(0) = \alpha$ and end at $\theta(t_f) = -\alpha$, where α is half the angle between the legs, and t_f is the time at which the next leg contacts ground. Use the starter code `rimlesswheelstance_startercode`. An event function will be needed to terminate the simulation. Use parameter values $\alpha = 0.3$, $r_{\text{gyr}} = 0$, $\gamma = 0.04$.



A. Demonstrate your simulation, starting at initial speed $\dot{\theta}(0) = -0.4$. Plot the states vs. time, as well as the total mechanical energy vs. time. Use the latter to devise a simple unit test, checking for energy conservation throughout the simulation.
B. Sketch the trajectory of the states for a stance phase in state-space, $\dot{\theta}$ vs. θ, along with an array of state-derivatives indicated with arrows. Note that the ground slope causes the

vector field to differ slightly from the normal inverted pendulum on level ground.

C. The rimless wheel experiences a collision with ground, such that the angular velocity is altered: $\dot{\theta}(0+) = \dot{\theta}(0-)\cos(2\alpha)$ where $0-$ and $0+$ denote the time instances immediately before and after impact. After impact, the next leg's stance phase commences, with a new initial angle α. On your state-space diagram, indicate these two steps, and show how the entire stance phase, collision equation, and switch to a new leg all can yield a periodic trajectory.

Due 25 January 2019