Scaling Analysis for Rigid Body Simulations

Andrew P. (Drew) Sabelhaus
apsabelhaus@berkeley.edu

Started 14 July 2014
Current as of 27 Aug 2017

This short white paper illustrates the scaling analysis that needs to be done on an rigid body simulation system, such as the NASA Tensegrity Robotics Toolkit (NTRT or its C++ library, NTRTsim) and the Bullet Physics system. This work is designed to convince NTRT/Bullet users that care must be taken for scaling techniques, moreso than occurs on the Bullet Physics forums or wiki. In particular, simple scalings work in some circumstances with certain types of simulated parameters, but may have to be modified elsewhere (see: scaling of forces, scaling of angular vs. longitudinal spring constants.) First, we introduce a simple example system for analysis, then do the nondimensionalization of parameters, and evaluate which scalings need to occur. Then, simulations of the example system are shown under these scaling conditions. Once this has been established, the anaysis is put in the context of NTRT simulations, and we show why simple simulations have (so far) worked in NTRTsim and where they may become improperly scaled in future work. Finally, suggestions are made for generalizing these concepts to new parameters in the simulator.

1 Example system, angular spring/damper

The first system we'll use as an example is not one that is currently used in NTRT. Though this doesn't help current NTRTsim users with their scalings, it is used to illustrate the potential pitfalls with scaling. Since an angular system is a completely reasonable model, this analysis shows that even a simple system model produces many (unintuitive) scaling requirements. See later sections for applications to the current NTRTsim model(s), once you've convinced yourself that this dimensionless term analysis is necessary.

In this example system, there is a rod on a hinge (hinge at (x,y) = (0,0)) with length l. There is an angular (rotational) spring, and an angular damper, acting at the origin also. Gravity is present, and we assume it acts fully at the center of mass for the rod.

With the state variables for this system as the angle between the rod and the horizontal, and its derivative,

$$x = [\theta, \dot{\theta}]^\top$$

We can derive the following system dynamics. (Note, these equations are not needed for scaling analysis, but are just provided for reference in case you'd like to simulate this yourself to check this math. I did!)

$$\sum T = I\ddot{\theta}, \quad \sum T = T_s - T_d + T_g$$

...where these torques are for the angular spring and damper, and the gravitational force. Their magnitudes are

$$T_s = k_\theta(\theta - \theta_r), \quad T_d = c_\theta\dot{\theta}, \quad T_g = (mg)(r) = \frac{mgl}{2}cos\theta$$

Here, θ_r is the rest angle of the spring. So, then, our rigid body dynamics are

$$\ddot{\theta} = \frac{1}{I}(\frac{mgl}{2}cos\theta + k_\theta\theta - k_\theta\theta_r - c_\theta\dot{\theta})$$

For a solid rod,

$$I = \frac{ml^2}{3}$$

And the full dynamics equation is

$$\ddot{\theta} = \frac{3}{ml^2}(\frac{mgl}{2}cos\theta + k_\theta\theta - k_\theta\theta_r - c_\theta\dot{\theta}) \tag{1}$$

Note that this system has a nonlinear forcing function: the gravitational force has a $cos(\theta)$ term in it. Although the plant can be formulated as a linear system, the whole system response cannot be. Thus, unlike the example linear systems in the last sections, arguments about transfer function magnitudes do not apply here in the same way.

We will not return to these equations of motion for the rest of the analysis, but it's useful for reference.

2 Π Terms and Nondimensional Numbers for example angular system

For an in-depth discussion of the necessity of Pi terms, and their use, history, and applications, see references such as chapter 7 of the undergraduate fluid mechanics textbook by Munsen [2] and Buckingham's original work on the subject [1]. It's sufficient to recount the following theorem, the Buckingham Pi Theorem, which states, loosely:

Given a system with n variables and parameters, of which there are k underlying basic units (phenomena), construct exactly $n - k$ nondimensional numbers. Then, if the magnitudes of each of these dimensionless numbers is kept the same under scaling of some of the parameters, via scaling other parameters equivalently, then the kinematic and dynamic responses of the original system and the scaled system will be *similar* (e.g. equal.)

Usually, n includes both constants in the system as well as anything else that one might care about, such as initial conditions or output results (forces, trajectories.) Also, for physical rigid body dynamics, k is almost always the set of $\{Mass, Length, Time\}$ or M, L, T, so $k = 3$ moving forward.

2.1 System variables and units

The above example system has the following parameters and variables (both implicit and explicit) and the following units. Here, we raise each of $\{M, L, T\}$ to the power of the underlying unit. For example, units of kg/sec^2 are represented as M^1T^{-2}. Note, though, that the following analysis does NOT imply the use of SI units, or American units, or any other unit system: it is valid for any unit system, as long as consistent choices are made for mass, length, and time.

$$\text{mass:} \quad m \sim M^1$$

$$\text{spring rest angle:} \quad \theta_r \sim 1, \text{ because we are using radians!}$$

$$\text{length of rod:} \quad l \sim L^1$$

$$\text{gravitational acceleration:} \quad g \sim L^1T^{-2}$$

$$\text{angular spring constant:} \quad k_\theta \sim M^1T^{-2}L^2$$

$$\text{angular damping coefficient:} \quad c_\theta \sim M^1T^{-1}L^2$$

$$\text{time:} \quad t \sim T^1$$

Note that since the angle here is in radians, and is unitless, the angular spring constant must be in units of torque.

These can be used to derive the units of force and torque as used in equation (1). In this formulation, force and torque are called *derived* units, separate from the *base* units of mass, length, and time. In other words, the units of force and torque are combinations of the other three, they can be defined fully in terms of the others. Specifically, they are the following:

$$\text{force:} \quad F \sim M^1T^{-2}L^1$$

$$\text{torque:} \quad \tau \sim M^1T^{-2}L^2$$

For example, the SI units here are $F \sim \text{Newtons} = \frac{kg*m}{s^2}$, and $\tau \sim N * m$.

This makes a total of 9 variables for us to consider scaling here. One question to ask might be - do we really need to scale force (or torque) since neighter unit appears in equation (1)? Equivalently - if we can't see it, do we need to consider it? We'll answer that question below. (Phrased another way: would we need to scale force/torque BEFORE solving the dynamics equations?)

Note that, unlike with longitudinal springs and dampers, their angular counterparts DO have a length dependence! This becomes important in a minute...

2.2 Generation of Π-Terms

In order to make use of Buckingham's Pi Theorem to scale our system correctly, we need to make $n - k = 9 - 3 = 6$ pi terms. I won't do that many until needed for illustrating a concept - I'll show the errors this generates as they occur.

The first step in this procedure is to select $k = 3$ "repeating variables," which will make up our pi terms. Any subset of 3 is fine here, including the base units if we'd like, under the following conditions:

1. The set of repeating variables must not be able to make a nondimensional set by themselves (i.e. they must be an independent basis)

2. Avoid selecting difficult-to-scale variables as repeaters (e.g. don't choose time here because that's difficult to scale dynamically on-line as the simulation runs.)

Let's choose, in this example, our repeaters as m, l, k_θ. See the Munsen textbook (or other resources) to see how to procedurally derive all the pi terms. Here are the first 3 of the 6 terms that would be needed. You can check the units in all of these: they divide out evenly.

$$\Pi_1 = \Pi_{\theta_r} = 0$$

$$\Pi_2 = \Pi_g = \frac{gml}{k_\theta} \sim \frac{(L^1T^{-2})(M^1)(L^1)}{M^1T^{-2}L^2} \sim 1$$

$$\Pi_3 = \Pi_{c_\theta} = \frac{c_\theta^2}{ml^2k_\theta} \sim \frac{(M^1T^{-1}L^2)^2}{(M^1)(L^2)(M^1T^{-2}L^2)} \sim 1$$

What we'll show now is that only doing 3 terms, and not the full 6, leads to incorrect results! Specifically, we'll show that if we neglect some of the parameters in the simulation (or forget about them), our results come out wrong.

2.3 Π-Term Analysis for Scaling

As an example, say that our simulation environment requires that we make a robot larger than its true dimensions. (This is the case with the NASA Tensegrity Robotics Toolkit and the underlying Bullet Physics.) For example, if we want to make all the lengths larger by a factor of s, we would write this as:

$$l_2 = (s)l_1$$

Intuitively: if the robot should be 10 times larger in simulation than in hardware, all the lengths that define the robot's mechanisms should be 10 times larger.

Then, we would need to choose a scaling of the remaining parameters such that all the pi terms remain unitless. Looking at Π_2 - how can we make it equal 1 still? Let's scale the gravitational constant of our simulation, g. This would be the following:

$$g_2 = (\frac{1}{s})(g_1)$$

These two scalings combined will keep Π_2 unitless. Next, we also need to keep Π_3 unitless. Since we've already specified changes to l and g, it would be easiest to scale

$$l_2 = \frac{1}{s} l_1, \qquad c_{\theta_2} = \frac{1}{s} c_{\theta_1}$$

However, there are other combinations we could choose! Pi term scalings are not unique. One could choose other sets of scalings but I'll leave that as an exercise for the reader! By the theorem, these scalings should all give us the same result.

3 Simulations of example angular spring system

In order to verify that following these scalings is actually required, here are a handful of simulations.

Figures 1 and 2 shows simulations of the original system, under parameters defined in the accompanying MATLAB file, as well as one with only gravity scaled by a constant. As expected, the latter figure shows an inherently different result. Here, the scaling factor was $s = 2$.

Figure 3 shows a simulation with length scaled with gravity. This one is done with proportional scaling: gravity multiplied by s, length multiplied by s. Note that this is similar to the assumptions passed around on the Bullet Physics forums: when scaling gravity, the only other scaling that's needed is length. Both can be seen as incorrect scalings.

Then, figure 4 shows the same system but this time following the three Pi-terms above. Specifically, all of g, l, and c_θ are scaled.

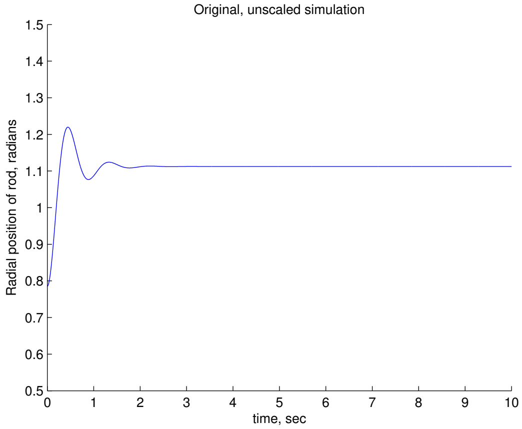

Figure 1: Original system, no scaling.

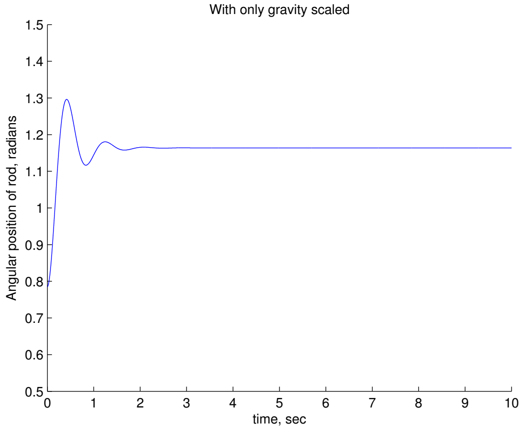

Figure 2: System with gravity scaled by 2.

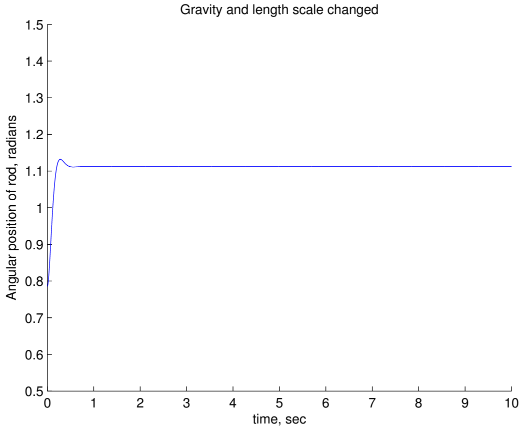

Figure 3: System with length scaled as well as gravity, both proportional with s.

What happened? The plots of Figures 2 and 3 are wrong, as expected, but so is Figure 4! It's not the same as Figure 1!

As designed by this tutorial, plot 4 shows what happens when not all Pi-terms are calculated or followed. We're not following the theorem exactly! We conveniently left out the Pi-term corresponding to the time scale. It's possible that this requires scaling also. Let's see if including this compels us to change our scalings or to scale additional parameters. After some algebra, this Pi-term is

$$\Pi_4 = \Pi_t = \frac{t^2 k_\theta}{ml^2}$$

Ack! The time scaling! This gets complicated now: ideally, we'd want to find a scaling that doesn't require a change to the time axis - this makes real-time simulations particularly difficult to interpret. However, a brief bit of investigation leads us to believe that any possible

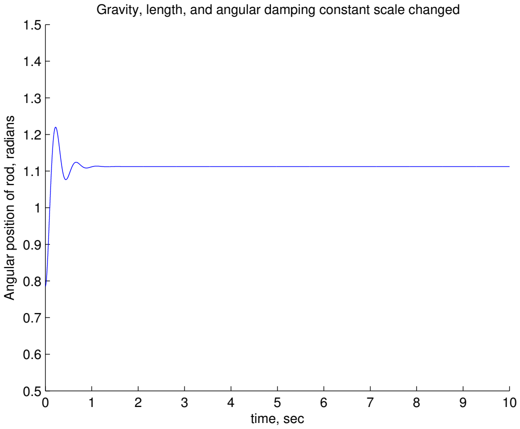

Figure 4: System with gravity, angular damping constant, and length scaled.

scaling for this system that involves a gravity term will also necessarily involve a time term. A future task is to show that this is true by either exhaustive search of Pi-term options or by finding other work that would show it as such.

Here's our final scaling with time:

$$l_2 = \frac{1}{s} l_1, \qquad c_{\theta_2} = \frac{1}{s} c_{\theta_1}, \qquad t_2 = (s) t_1$$

This system with all four terms scaled is simulated in figure 5. See how this is the same result as figure 1, finally!

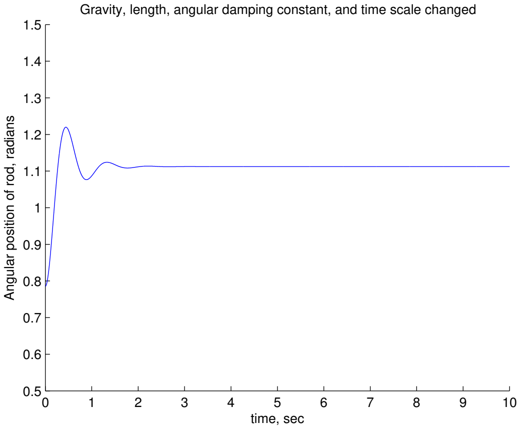

Figure 5: System with gravity, angular damping constants, and time scaled.

3.1 Discussion

This section should show by contradiction that the following two statements are untrue:

1. Commutativity of multiplication implies that we only need to scale length when scaling gravity, e.g., only scaling length will result in similitude. $n * (length/time^2) == (n * length)/time^2$.

If this were true, not only would figures 1 and 3 be identical, but there wouldn't be a need for dimensional analysis in the first place. This conjecture relies on a notion of commutativity of multiplication for a single equation - that involving only length and gravity - and neglects equations that might involve gravity and other terms, or length and other terms. This is the motivation behind the Buckingham Pi Theorem: such a procedural method ensures that all such interactions are accounted for, and the many potential multiplicative commutativity relationships are always satisfied.

2. Instead of simulating a robot of characteristic size l, mass m and gravity g, we simulate a robot of size 10*l, mass m and gravity 10*g. The result will be dynamically and kinematically similar (in the technical sense of similitude.)

Similar to the above reasoning, this neglects other possible variables in the system. While it is true that such a scaling is needed, this conjecture doesn't consider other possibly-necessary scalings. In particular, even something like mass may need to be scaled here depending on the situation and if the Pi-terms call for it. Care must be taken! However, note again that this does work for certain linear systems, such as those described with the simpler Pi-terms in section

4 Analysis of related longitudinal spring/damper systems, for NTRT use

The above analysis is based on a system where parameters are angular. However, our simulated systems use longitudinal/linear dampers and springs, along with some other parameters, not angular systems. The analysis changes quite a bit here! This is somewhat unexpected at first - why would the units of these extra parameters influence the required scaling for unrelated terms? - but by the Pi theorem, it must work this way. As alluded to below, this change in scaling requirements arises from the inclusion of a spring constant term that does not depend on length.

Ignoring a full dynamics derivation (no more simulations for me!), let's take the following constitutive equations for longitudinal springs and dampers:

$$F_s = k(x - x_r), \quad F_d = c\dot{x}$$

Here are the units of all the CONFIG parameters for the most recent SUPERball model in NTRT.

$$\text{gravitational acceleration:} \quad g \sim L^1 T^{-2}$$
$$\text{density:} \quad \rho \sim M^1 L^{-3}$$
$$\text{linear spring rest length:} \quad x_r \sim L^1$$
$$\text{linear spring constant:} \quad k \sim M^1 T^{-2}$$
$$\text{linear damping coefficient:} \quad c \sim M^1 T^{-1}$$
$$\text{pretension force:} \quad F \sim M^1 L^1 T^{-2}$$
$$\text{target actuator velocity:} \quad v \sim L^1 T^{-1}$$
$$\text{time:} \quad t \sim T^1$$

Additionally, some models in NTRTsim have other parameters, such as Brian Mirletz' Corde model that uses a shear modulus and modulus of elasticity:

$$\text{shear modulus:} \quad G \sim M^1 L^{-1} T^{-2}$$
$$\text{modulus of elasticity:} \quad E \sim M^1 L^{-1} T^{-2}$$

Note the missing length scale in k and c. This is because, for example, k needs to be units of N/m instead of N/rad, and the extra length term cancels out. The new Pi terms, using the same set of $\{m, l, k\}$ are:

$$\Pi_1 = \Pi_{x_r} = \frac{x_r}{l}$$
$$\Pi_2 = \Pi_g = \frac{gm}{kl}$$
$$\Pi_3 = \Pi_c = \frac{km}{c^2}$$
$$\Pi_4 = \Pi_F = \frac{F}{kl}$$
$$\Pi_5 = \Pi_t = \frac{t^2 k}{m}$$
$$\Pi_6 = \Pi_G = \frac{Gl}{k}$$
$$\Pi_7 = \Pi_E = \frac{El}{k}$$
$$\Pi_8 = \Pi_v = \frac{v^2 m}{l^2 d}$$
$$\Pi_9 = \Pi_\rho = \frac{\rho l^3}{m}$$

Note that here, the scaling of gravity with length does satisfy almost all of these Pi terms. When g is multiplied by s, we can do the same to l. That then would require scaling by

s to F and v, scaling by $1/s$ to G and E, and by $1/l^3$ to ρ. So, time does not need to be scaled here, but still, five other parameters beyond just gravitational acceleration need to be scaled. The takeaway here is that without extra Pi terms for more parameters like k_θ, we can get away with this simple scaling, but we need to be careful and re-evaluate our Pi terms each time a new unit parameter is introduced.

5 Scaling Analysis for General Linear Systems

Finally, I'll finish with an informal argument as to why a truly linear system would not need any of this scaling analysis. In brief, if input forces can be represented as time-invariant, and the Laplace transform of the system shows a scale of the input to correspond proportionally to an output magnitude change, then scaling is simple and intuitive. I originally tried these simulations using a simple spring-mass-damper vertically oriented under gravity, e.g.

$$m\ddot{x} + c\dot{x} + kx = mg$$

With this system, the "gravity proportional to length" scaling also works, but for an entirely different reason. What's actually going on in this case is a change in the magnitude of the input-output transfer function, and so the scaling ITSELF is non-dimensional! One way to see this is to notice that the gravitational force is time-invariant, and could thus be replaced by a step input:

$$m\ddot{x} + c\dot{x} + kx = u(t)$$

By superposition of linear systems, the magnitude of the output of the system of

$$m\ddot{x} + c\dot{x} + kx = (n)u(t)$$

...will be exactly n times larger. Thus, dividing the output by n re-scales to the proper dimensions. This is, however, informal: more research must be done to show this result, and to review the literature for similar work or proofs.

6 Moving Forward

So, after all this work, what should a simulation user do?

First, review one's model and ensure that the model's scalings are appropriate. If your model parameters only have the units above, then following the corresponding Pi terms I derived should be enough. However, if your model either does not include a linear spring constant k or involves other terms not analyzed here, you must do a new dimensional analysis for your results to be correct (e.g. dynamically similar.)

Additionally, simulation users must be mindful of parameters with units inside their controllers, not just in their models. The Buckingham Pi Theorem only works if all terms in the entire system equations are taken into account, and with a closed-loop controller, the controller dynamics must be included. As one example, developing a model-based controller with parameters calculated analytically would have to be scaled. A controller using nonlinear damping techniques would require a complete re-derivation of these Pi terms and scaling.

Please feel free to use the MATLAB code in the subdirectory here to help analyze your system, if needed. Email me if you find issues.

References

[1] Edgar Buckingham. On physically similar systems; illustrations of the use of dimensional equations. *Physical Review*, 4(4):345–376, 1914.

[2] Bruce Munson. *Fundamentals of fluid mechanics*. John Wiley and Sons, Inc, Hoboken, NJ, 2013.

7 Ongoing Questions

1. Do we need to include 'invisble' terms, like force/torque in the first example?

2. If our system uses a unit like torque, that depends on force, do we just choose one of the two? E.g. just choose force, then no need to scale torque? Or do we need to choose both?

3. Can we use the base units as the repeating variables? E.g., is it valid to use mass and length as repeating variables when we use mass, length, time as the base units?

4. When we 'scale' torque, does that mean we need to scale the applied torque every place that the torque ie applied? E.g., do we need to scale BEFORE solving the dynamics equations? E.g., when doing a numerical simulation, do we need to scale the applied torques before they are actually applied?

5. Can we scale the base units? Would we ever scale mass, length, or time here? Or does needing to do that imply that something is wrong in the analysis? Or is it OK, but just makes our simulation more difficult to implement?