Using Kangaroo (Grasshopper version)
(DRAFT)

(you can download this as a pdf, but then you won't get the animations)

Contents

Intro

Kangaroo is an add-on for Grasshopper/Rhino and Generative Components which embeds physical behaviour directly in the 3D modelling environment and allows you to interact with it 'live' as the simulation is running.
It can be used for various sorts of optimization, structural analysis, animation and more.

This guide assumes you are already reasonably competent in Rhino and Grasshopper. If you aren't don't worry though, they are both pretty intuitive and easy to pick up the basics of, with plenty of good online documentation and support available:
http://www.rhino3d.com/support.htm

http://www.grasshopper3d.com/

Author - Daniel Piker
Development team - Robert Cervellione, Giulio Piacentino, Daniel Piker

Big thanks to:
David Rutten, Bob McNeel, Andrew Payne, Luis Fraguada, Moritz Fleischmann, Jaryd Carolin, Rabindranath Andujar, Steve Baer, Chris Williams, Sam Joyce, UTO, Studio Mode, Gennaro Senatore
(loads more to add here, sorry if I've missed you out)

The development of Kangaroo is currently not supported or funded by any company or institution.
It is shared freely in the hope that you will enjoy it and use it for original and creative work.
It is free to use for academic and commercial purposes.
Attribution and feedback is always appreciated.

Disclaimer

Please bear in mind that Kangaroo is still very much a work in progress, which you use entirely at your own risk. While every effort has been made to make its behaviour physically correct, no guarantee of accuracy is given.
The developers are under no obligation to provide support.

Getting started

You have presumably already downloaded Kangaroo itself. At the time of writing the latest public release is 0.044. Links to any future updates will be posted at kangaroophysics.com

You will also need an up to date version of Rhino 4 (currently Service Release 8)
or the latest Rhino 5 beta.
If you do not already own Rhino you can still use Grasshopper and Kangaroo by downloading the fully featured trial version from http://www.rhino3d.com/download.htm.
You need to have the Grasshopper plugin installed. It is free and you can download the latest version from http://www.grasshopper3d.com/.

To install Kangaroo, simply move the .gha file into the grasshopper components directory
(typically C:\Program Files\Rhinoceros 4.0\Plug-ins\Grasshopper\Components)
and restart Rhino.
Alternatively, to avoid having your custom components overwritten whenever Grasshopper auto-updates you can create a separate directory somewhere else on your hard disk then type GrasshopperDeveloperSettings from the Rhino command line and add a search path to that directory.
Kangaroo 0.044 is not compatible with files created with pre 0.04 versions of the plug-in.

What is a particle system ?

> "Particles are objects that have mass, position, and velocity, and respond to forces, but that have no spatial extent.. ..Despite their simplicity, particles can be made to exhibit a wide range of interesting behavior. For example, a wide variety of nonrigid structures can be built by connecting particles with simple damped springs."
> [Witkin 97]

The particles we deal with in Kangaroo are an abstraction, but one with a strong connection to our understanding of how the real world works at a fundamental level.
Macroscopic properties of materials such as their behaviour in bending, shear and torsion can actually be seen as ***emergent*** on a molecular level from simple interaction between pairs of particles.

Of course in the real world, objects are made out of vastly more particles than we use in simulation, but if some care is taken about how we distribute the points and their masses we can get quite good approximations of real physical behaviour using this method.

While they do have their limitations, one great advantage of particle systems is that they are easily understood and controlled (compared with more sophisticated continuum models).
This conceptual simplicity makes it possible for designers to apply and manipulate the physics simulation in a very direct way, without needing specialist technical knowledge.

(by the way, the terms ***particle, node*** and ***point*** are used fairly interchangeably in this guide)

The main Kangaroo engine component

To start creating a simulation you must place the main Kangaroo component on the canvas.
(After installation and restarting Grasshopper all Kangaroo components should appear under the 'Extra' tab).

You only need one instance of this component per definition.

Remember that hovering over a component without clicking will show a very brief description of what it does, and hovering over each of the inputs or outputs will display further information.

- Force Objects

Kangaroo contains various ways of generating forces which affect the particles in the simulation. All of these are fed into the 'Force Objects' input.
Forces can come from various sources - the response of materials to deformation (elasticity), user input, geometric constraints...
By treating all of these within the common language of force vectors Kangaroo allows live interaction between them.

The input to Force objects needs to be a flattened list. (To flatten the input right click on the name and select Flatten).
All the different possible types of forces are covered in more detail later in this guide.

- Anchor Points

These can be used to keep points fixed in a certain location. No matter what forces are applied to them they will not be moved by Kangaroo. However, you can still move them in Rhino to interact with the simulation as it is running.

Particle consolidation - Joining objects together

If two or more points in the initial input (Force Objects and anchors) are in the same position (to within the tolerance setting) they will get joined together and treated as a single particle by Kangaroo.
There is no need to specifically weld objects to each other.

Anchor points which start out coincident to points of Force Objects (such as the end of a spring) will therefore move those objects around with them when you move them during the simulation. Bear in mind though, if you stop the simulation and Reset it, if the anchor points no longer coincide with the points of the force objects they will no longer be connected. So you may sometimes need to undo your movements of the Anchor points in Rhino before running the simulation again.
(add illustration)

For now if you do want to have to objects start in the same place but not get joined you must move them apart by a small amount. (illustration)
If there is demand for it I could add an 'unweld' component to get around this.

- Settings

There are a number of global settings for the simulation which can be accessed from the 'settings' input of the main Kangaroo component.
You can access these settings either by adding a settings component to the canvas (the spanner/ screwdriver icon), or via the rollout menu when you right-click the settings input (though watch out - settings adjusted in this way are currently not saved with your ghx definition).

-Tolerance
The minimum distance between separate points. Points closer than this will be consolidated into one.

-TimeStep
How far through time the system moves at each iteration. Smaller values will result in a more stable, but slower simulation. Stronger forces and stiffer springs require smaller timesteps.

-SubIterations

The number of iterations calculated between each time the solution is drawn on screen.

-Floor
A simple constraint which can be switched on or off preventing particles moving below Z=0
This is much faster than Brep collision.

-Drag
A force on all particles resisting their motion.This is essential to make a system settle down to a static equilibrium position. If drag is too low, the system will oscillate for a long time, if it is too high, the particles will move slowly.

-Restitution
How elastic collisions between particles and the floor are. If 1 then particles bounce back to the height from which they were dropped, if 0 they do not bounce at all.

-StaticFriction
-KineticFriction

-Settle

-Tumble
How much horizontal velocity is conserved in collisions between particles and the floor.

-Sound
Sound effects. This feature is disabled in the current version.

-Solver
The integration method used by Kangaroo to calculate new positions for the particles.

Utilities

Kangaroo contains a remove duplicates tool :
 which can be used to clean up lists of points. It has a tolerance setting which you can set to some small value (such as 0.01) so that any points closer than this will be combined into one.

RemoveLines

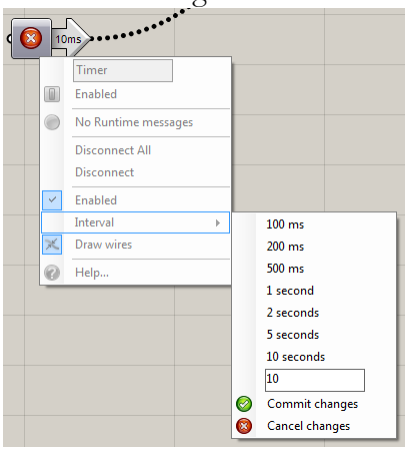
InterConnect

Running the simulation

To start and stop the simulation you need to attach a Toggle (Parameters>Special>Boolean Toggle) to the **SimulationReset** input of Kangaroo. When this is set to True Kangaroo does some initial

pre-calculation and matches all the inputs to the appropriate points. When it is set to False the simulation moves forward one iteration every time the grasshopper solution updates.

To make the simulation continuously update you need to attach a Timer component (Parameters>Special>Timer). Drag the dotted line from the timer to any part of the Kangaroo component. The timer has an **interval** setting which controls how long it waits between updating the solution. Right click and change this interval to 1ms to get the maximum speed.



Displaying the Grasshopper canvas takes quite a lot of memory and slows down the simulation considerably. To avoid this you can minimize it while running your simulation.
Instead of double-clicking the Timer component to turn it on and off you can double click the global toggle which will appear on your Taskbar the first time you use the Timer.

Newton's Laws

The change of momentum of a body is proportional to the impulse impressed on the body, and happens along the straight line on which that impulse is impressed

(illustration)

This law is probably most familiar to you as

$$\mathbf{F} = \mathbf{ma} \qquad (1)$$

force = mass × acceleration

Which we rearrange to get $\mathbf{a=F/m}$ (2)

Kangaroo works by finding the total force vector F for each particle by adding up all the different forces acting on it, using Newton's second law to get the acceleration, and numerically integrating the resulting differential equation of motion over time to find new positions for all the particles.

Note that in Kangaroo ***mass*** is not automatically associated with ***weight***.
More **mass**ive objects take more force to change their velocity.
Weight is the force on an object due to gravity.
In real life an object's weight is proportional to its mass.
In Kangaroo you can easily make this the case, or you can assign them completely independently (for example in order to test specific loading cases).

It is possible to assign a particle zero weight, but giving a particle zero mass would result in a division by zero in equation (2) above and cause an error.

You do not always have to specify values for mass - if no specific input is provided, all particles will be assigned a default mass of 1.
To assign a particle a particular mass, or give it an initial velocity, use the Particle component.

To include the effects of gravity and give a particle weight, you must apply a ***Unary force*** to it.

(illustration)

To every action there is always an equal and opposite reaction

(illustration)

This is why Kangaroo uses lines to input forces. Each force is an interaction between a pair of particles - the endpoints of the line. The force acts in the direction of the line and is added to one particle and subtracted from the other.

Unary forces are an exception to this - they only apply to single particles. However, what we are effectively doing here is pairing the single particle with another infinitely distant and infinitely massive particle - a reasonable approximation to the centre of the earth for many practical purposes.

Every body persists in its state of being at rest or of moving uniformly straight forward, except insofar as it is compelled to change its state by force impressed

(illustration of vector summation)

If the forces on a particle sum to zero then it will not accelerate.

Conversely, while the forces do *not* sum to zero, the particle's velocity will continue to change.
If we include in our simulation some form of ***friction*** or ***drag*** then all particles will continue to move until they eventually settle in an **equilibrium** position where the net forces on each particle sum to zero.

(I will eventually add a total KE output. We could then use this to tell the simulation to run until it drops below some given threshold.)

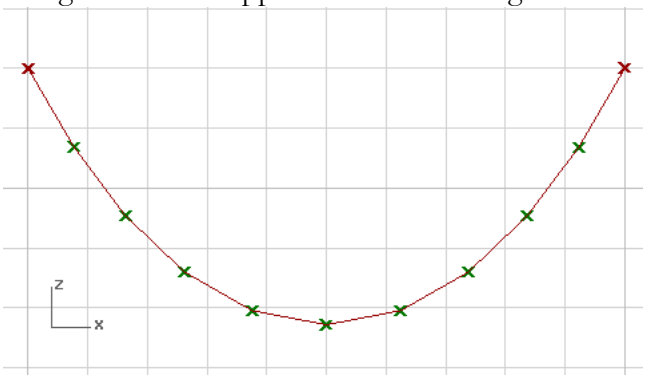
Discretization

Because the geometry of a spring in Kangaroo is determined by just 2 points, it can only ever be a straight line.
To make **flexible** elements we must first **break them up into smaller pieces** and model each segment as a separate spring.

Here we make certain simplifications, for example, when modelling a hanging chain we treat it as though all the mass is concentrated at the ends of each segment, when in reality it is distributed over its length.
However, provided some care is taken with how the masses and forces are assigned then these ***lumped mass*** models can often provide a decent approximation of continuum mechanics (even when quite coarse subdivision is used).

Cables

Cables can be divided up into segments either in Rhino or in Grasshopper
(illustration)
Dividing the curves in Rhino might be easier for those less comfortable with Grasshopper, but doing it in Grasshopper can make altering the definition easier.



a flexible cable is modelled as made up of multiple straight segments

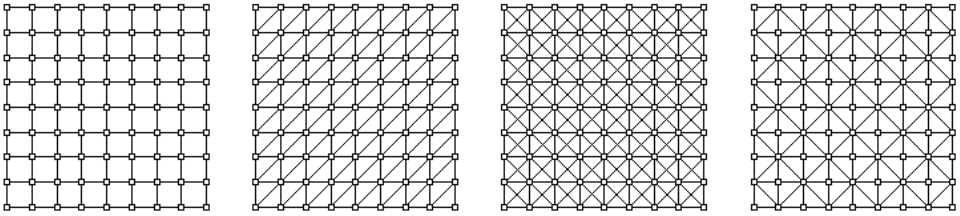
Sheet materials - membranes, fabrics, paper etc.

The simplest way to make a sheet material in Kangaroo is to use a grid of springs.
Weaverbird (downloadable from http://www.giuliopiacentino.com/weaverbird/) has some very useful components for extracting the edges of meshes as curves. UTO's mesh tools (http://utos.blogspot.com/p/downloads.html) are also very useful.

A grid of zero rest length springs will behave something like a **soap film** - trying to minimize area. So if you don't restrain the edges it will shrink down to nothing. If you wanted to avoid this (for

example if you are simulating tensioned fabric), you could use a rest length for each spring which was some multiple between 0 and 1 of it's start length (illustration). You could also increase the stiffness of springs around the outside of the mesh to simulate edge cables.
For catenary structures where you want to simulate a hanging grid of chains you may want to make the rest length *more* than the start length to give it some slack.

To more realistically model the behaviour of **cloth** you may also want to add ***shear springs*** - diagonals which stop each square from deforming into a diamond shape.
(see http://vimeo.com/15022048)
Different stiffness values can be used for the main square grid springs and the shear springs for different cloth properties.
There are a number of choices for how these diagonals are added - 1 or 2 per quad, aligned or alternating. I haven't tested much yet how this choice affects the behaviour.



You can also add bending stiffness - (useful for some cloth behaviour and particularly for sheet materials like paper or metal). The typical way to do this is to connect springs between alternate points in the grid, but a better way is to use the bending elements described in the section below on rods.

Rods

Bending resistance works with sets of 3 points. It will try to keep those 3 points in a straight line. So to model an ***elastic rod*** you need to input the 1st 2nd and 3rd nodes, the 2nd 3rd and 4th nodes, etc (illustration)
To fix the **tangent** of a rod simply make 2 points at the end anchor points. To just fix the end position only use one anchor point.
There is currently no way to simulate resistance to torsion for 1D elements in Kangaroo, so it is as though the rod is free to twist about its own axis at the constrained ends. (Though you can make solids with torsional resistance as described below).

Solids

(more to add here)

There are probably many more useful discretizations to be invented. Because this part of things happens in grasshopper outside of the kangaroo code, anyone has the control and freedom to explore this.

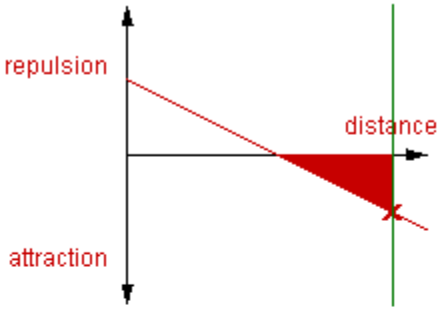
Springs

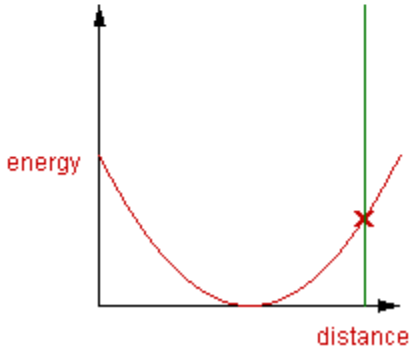
It might seem odd at first to simulate entire structures with springs - but we are not just talking about the kind of springs in car suspensions and mattresses. Even the stiffest materials stretch and compress when we apply forces to them.

Hooke's law says that the force exerted by a spring is directly proportional to the amount its length differs from its *natural* or *rest* length.
This is a simplification, but often a good approximation for many materials in ordinary use.

Note there is no Kangaroo input for the ***start length*** of a spring - it simply uses the length of the curve input to *springs*.
The ***rest length*** (also called natural or slack length) of the spring is the length it 'wants' to be.
If you do not supply a value for rest length it defaults to 0. Often you will want to make the start length the same as the rest length - which you can do by simply connecting the curve to the rest length input (illustration).
You can also include a multiplier, so that the rest length is some multiple of the start length. If this multiplier is between 0 and 1 it will be like pre-tensioning the spring. (illustration)

Spring



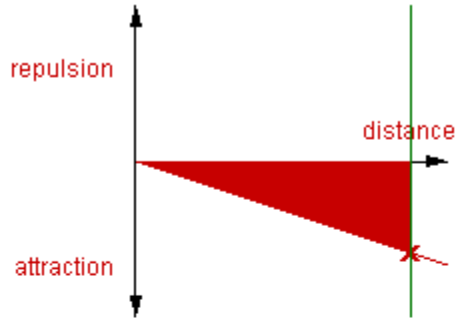


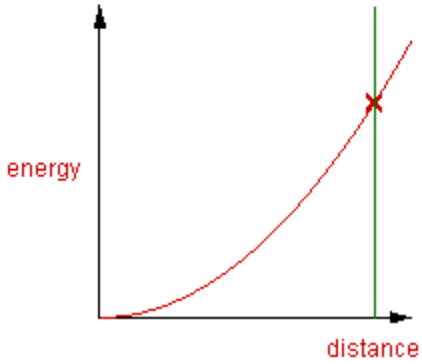
The top graph represents the force calculation that takes place within Kangaroo.

The distance at which the line crosses the horizontal axis is the natural length of the spring.
The stiffer the spring, the steeper the line.

The bottom graph shows how the ***potential energy*** varies.
Note that its speed(***kinetic energy***) is greatest when its potential energy is lowest and vice-versa.
Without drag it would keep sliding up the slope to the same height on the opposite side as energy was converted back and forth between potential and kinetic.
When drag is added to the system these oscillations decrease in amplitude until it settles in a position where the potential energy is ***minimized***. This is the essence of how physics-based ***optimization*** in Kangaroo works.

Zero rest-length spring

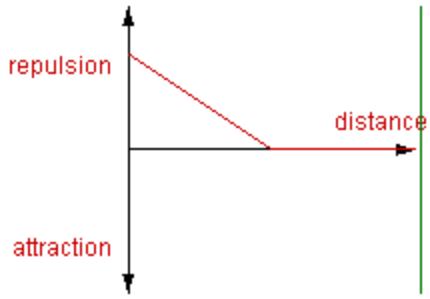


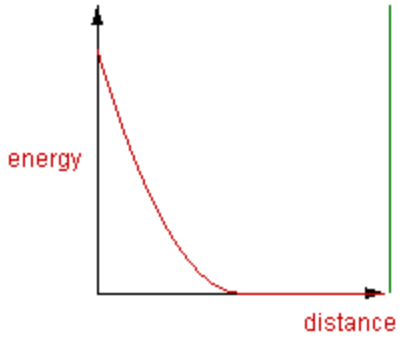


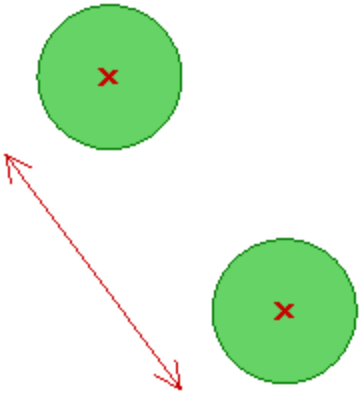
Springs with a natural length of zero are often useful. For example minimal surfaces can be roughly approximated by treating all the edges of a mesh as zero-length springs.

Cut-offs

Balls (Spring with positive Cutoff):

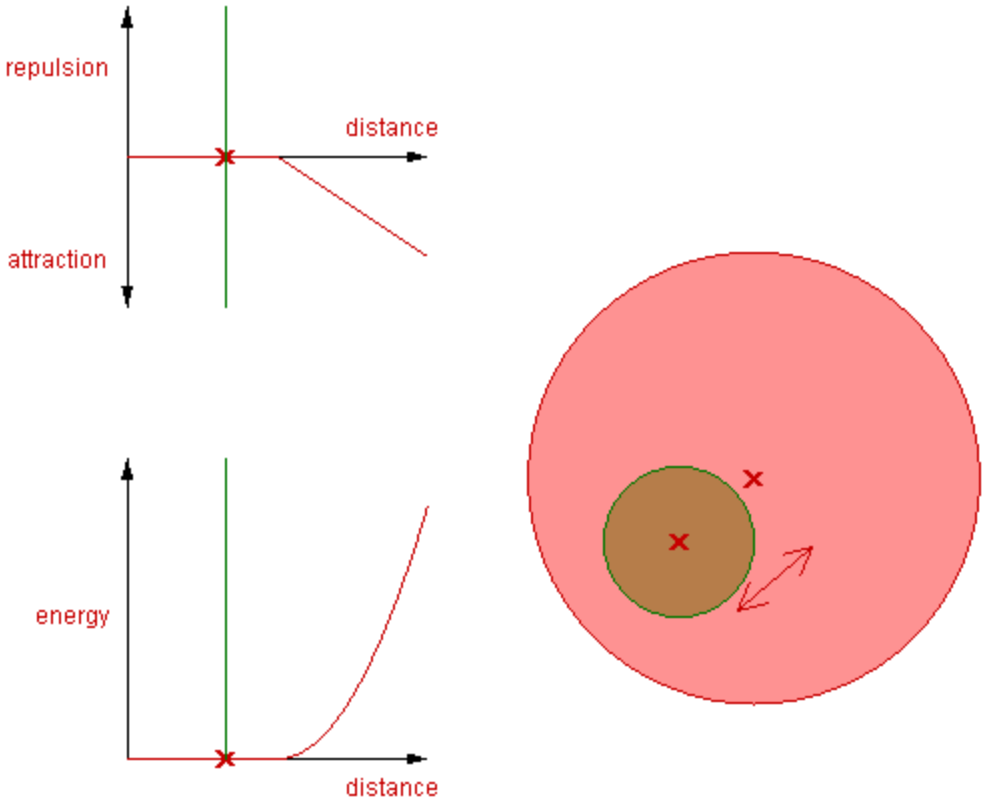






The SpringCutoffs input lets you specify a distance beyond which to ignore the effect of the spring. If you use the same value for SpringLength as for SpringCutoff, then the particles behave like a pair of solid balls which exert no force on each other when they do not touch, but push apart when they overlap (ie when the distance between their centres is ***below*** that value, which should be equal to the ***sum of the radii*** of the 2 spheres) .

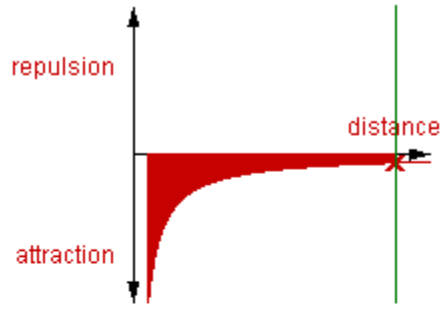
'Hamsterball' (Spring with negative Cutoff):

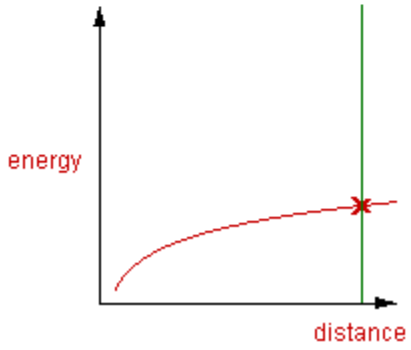


If the value you supply to SpringCutoffs is negative, then the force due to that spring will only have an effect ***above*** that distance. When you make the Cutoff value the negative of the SpringLength value, this is like one sphere being inside another, with the value equal to the ***difference between their radii***.

Cut-offs for Power Laws work in exactly the same way as for springs.
They can be particularly helpful when using power laws with a larger negative exponent (eg -3 or -4). When the force increases very steeply at short distances (see the section on Power Laws below) it can cause particles to 'explode' apart when they get too close. A small negative cutoff (say -0.1) can help avoid this.

Power Laws





This shows a Power law force with ***exponent* -1**, which means that the strength of the attraction is proportional to the ***inverse*** of the distance between the particles.

A basic math reminder:

$$x^{-n} = \frac{1}{x^n}$$

Using a more negative exponent (such as **-2**, the familiar ***inverse-square*** law) would mean a force which increased more sharply at short distances and fell off towards zero more quickly at larger distances. (illustration)

Power Laws in Kangaroo only accept ***integers*** as exponents- because raising a number to an integer power is much quicker to compute than raising it to an arbitrary rational number, and most known real physical forces have integer exponents.
*edit - why is this ? are there any known forces with non-integer exponents?

If the strength of a Power Law is ***positive*** then it is ***repulsive*** - it tries to ***maximize*** the distance between the particles.

If the strength of a Power Law is ***negative*** then it is ***attractive*** - it tries to ***minimize*** the distance between the particles.

Tip - When using attractive power laws with large negative exponents you may find it is useful to include a small negative Cutoff value. This is because otherwise at very short distances the force increases so sharply it can cause the particle to dramatically overshoot.

Other Forces
(descriptions coming shortly)

Planarization
Pressure
Constraining to a surface
Brep Collisions
Equalization
Rockets

Combining forces

* include something about combination forces such as Lennard-Jones here

Unary Forces

These have already been covered a bit earlier in the section on Mass and weight.
They can also be used to add other forces which are not dependent on the distance between particles, such as simple wind loads.

Drag

Drag is a force which acts on particles opposing their direction of motion. The faster the particle is moving, the greater the force.
Kangaroo has several sorts of damping :
Global damping - which is applied uniformly to all particles in the simulation. The 'Drag' input allows you to set the strength of this. Values between 1 and 30 seem to work well. You may find you need to experiment with different values to get the effect you want. For example when form-finding a catenary roof, if the damping is set too low, the form fall then bounce back up and oscillate for a long time before settling into the desired equilibrium. If damping is set too high it wall fall very slowly, as if through treacle.
Spring Damping - This force is a setting for springs, which also opposes the movement of the particles, but only in the direction of the spring. So for example a horizontally stretched spring with spring-damping allowed to fall freely under gravity will still reach the ground in the same length of time, although its length oscillations will reduce.

(Entropy, energy loss...)

Outputs

Explanation of Geometry Tree to go here

Troubleshooting

-The simulation oscillates wildly or explodes
Because Kangaroo is numerically approximating continuous* behaviour with discrete time steps, errors can occur. Usually these errors remain very small, but when dealing with very stiff springs, particles sometimes overshoot in a way that builds up rapidly, causing the whole system to fly apart. The solution is either to use softer springs or to decrease the timestep (try reducing it by a factor of 10, eg from 0.01 to 0.001). Increasing Drag or Spring Damping can also help. Smaller timestep means more calculation steps for the same amount of movement, so the simulation will run slower. One way to counteract this is to increase the value of SubIterations (typically by the same factor you reduced the timestep). This means that multiple calculations of the particles' positions occur between the ones which you actually see on the screen.

* Well it seems continuous, but is time really continuous or discrete ? That's a question for another time!

Other sections to add:

Tracing particles

producing animations (slider)

speedup tips (system process priority)

References/Further reading

Baraff & Witkin's Siggraph '97 course notes
http://www.cs.cmu.edu/~baraff/sigcourse/
http://www.pixar.com/companyinfo/research/pbm2001/

Physics-based generative design
Ramtin Attar, Robert Aish, Jos Stam, Duncan
Brinsmead, Alex Tessier, Michael Glueck,
Azam Khan
http://www.autodeskresearch.com/pdf/CF09PhysicsGenDesign.pdf
http://www.autodeskresearch.com/projects/complexconstraint
http://www.autodeskresearch.com/projects/nucleus

Physically based deformable models in Computer graphics
Andrew Nealen, Matthias Müller, Richard Keiser, Eddy Boxerman, Mark Carlson

http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.124.4664&rep=rep1&type=pdf

http://www.matthiasmueller.info/realtimephysics/

Jeffrey Traer Bernstein's Traer Physics
http://murderandcreate.com/physics/

lots more to add here

Future development plans

new forces to add:

vortex filament/ driveshaft/ hingespring
To rotate one particle about a line segment. Could be Biot-Savart like, or like PLaw where user has a choice of how distance affects strength. Useful for wheels, folding...

alignment
align 2 line segments which are not necessarily contiguous. could be used for flocking

bending with non-straight rest angle
eg to constrain 2 segments to be perpendicular

wind
area dependent like pressure, but projected onto a wind vector

air resistance
like wind, but opposed to the motion of the triangle

shear
2 line segments try to keep the same normal plane

plate/ shell elements ?
This is a tricky one - would really appreciate suggestions here...

breakable springs

Add/improve integration methods

Total Kinetic Energy output (and maybe an option to run the simulation until this falls below a certain level)

If you have any suggestions or requests for other forces to add, please let me know