This pdf is a rough overview of theorem, equations, and the algorithm applied in my Matlab code for a N-particle simulation of Newton's law of universal gravitation.

The original Matlab code (and origin of this pdf) can be found at:

https://github.com/JamieMJohns/N-Particle-Simulation-of-Newton-s-Universal-Law-of-Gravitation-Matlab-

There are many sources of information for the theorem/equations and algorithm some quick useful sources I found are;

http://theory.uwinnipeg.ca/physics/circ/node7.html

https://introcs.cs.princeton.edu/java/assignments/nbody.html

https://kof.zcu.cz/st/dis/schwarzmeier/gravitational_simulation.html

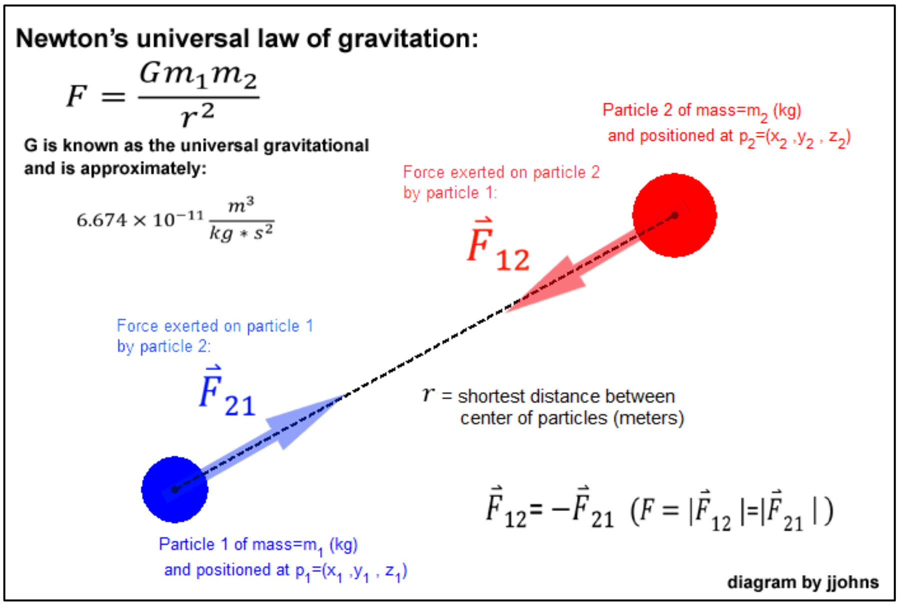
https://msdn.microsoft.com/en-us/library/dn528554(v=vs.85).aspx

And of course, Wikipedia;

https://en.wikipedia.org/wiki/Newton%27s_law_of_universal_gravitation

(pdf created by Jamie M Johns, 2018)

Newton's universal Gravitation law:



Vectorization:

$$\vec{F}_{ab} = \frac{Gm_am_b\vec{r}_{ab}}{r_{ab}^3} = \{F_{x_{ab}}, F_{y_{ab}}, F_{z_{ab}}\}$$

$$\vec{r}_{ab} = \vec{p}_b - \vec{p}_a = \{r_{x_{ab}}, r_{y_{ab}}, r_{y_{ab}}\}$$

$$= \{x_b - x_a, y_b - y_a, z_b - z_a\}$$

$$r_{ab} = \sqrt{{r_{x_{ab}}}^2 + {r_{y_{ab}}}^2 + {r_{z_{ab}}}^2} \quad , \quad r_{ab} = r_{ba}$$

Where, for example of $F_{x_{ab}}$; $\quad F_{x_{ab}} = \frac{Gm_am_b\,(x_b - x_a)}{r_{ab}^3}$

Newton's second law of motion (F_{net}=ma):

$$F_{net_i} = \sum_j F_{ji} \quad (j \neq i)$$

$$= m_i a_i$$

$$= m_i \frac{dv_i}{dt}$$

Finite difference approximation of a first derivative;

$$\frac{dv}{dt} \sim \frac{v(t + \Delta t) - v(t)}{\Delta t} \qquad (v = velocity)$$

$$a = \frac{dv}{dt} \qquad (a = acceleration)$$

Application of finite difference;

Let,

$$F_{net} = m\left(\frac{v(t+\Delta t)-v(t)}{\Delta t}\right) \qquad \{F_{net} = ma, \ \ a = acceleration\}$$

Rearrange the above to;

$$v(t+\Delta t) = v(t) + \Delta t \ \frac{F_{net}}{m} \qquad \text{(change in velocity)}$$

And, for change in position;

Let,

$$v(t+\Delta t) = \frac{x(t+\Delta t)-x(t)}{\Delta t} \qquad \{v = \frac{dx}{dt}\}$$

And rearrange for position at the next time step;

$$x(t+\Delta t) = x(t) + v(t+\Delta t)\Delta t$$

Definition of variables:

(x,y,z)=position in meters for three dimensions [distance from origin (0,0,0)] (Units:m)
t=time in seconds (Units: s)
m=mass (Units: kg)
v=velocity (Units: m/s)
a=acceleration (Units: m/s^2)
F=Force (Units: (kg*m)/s^2)

Algorithm:

For N sperate particles, each particle i is defined with a mass (m_i), position ($\vec{p}_i = \{x_i , y_i, z_i\}$) and initial velocity ($\vec{v}_i \ = \{v_{x_i} , v_{z_i}, v_{z_i}\}$, if the object is initial not moving then $\vec{v}_i = \{0,0,0\}$).

Delta time (Δt) is also defined.

At first step of t=0 (initial state) and for each particle i ;

$\vec{p}_i(t) = \vec{p}_i$, $\vec{v}_i(t) = \vec{v}_i$ (and time(t)=0)

For next time steps, t ≥ 2:

For each particle i:

Initialize; $\{F_{x_i}, F_{y_i}, F_{z_i}, \}$ ={0,0,0}

$F_{x_i} = \sum_{j=1}^{N} F_{x_{ji}} \quad \{for\ j{\neq}i\,)$

$F_{y_i} = \sum_{j=1}^{N} F_{y_{ji}} \quad \{for\ j{\neq}i\,)$

$F_{z_i} = \sum_{j=1}^{N} F_{z_{ji}} \quad \{for\ j{\neq}i\,)$

$v_{x_i}(t+1) = v_{x_i}(t) + \Delta t\, \frac{F_{x_i}}{m_i}$ {and similar for $v_{y_i}(t+1)$ and $v_{z_i}(t+1)$}
$x_i(t+1) = \ x_i(t) + \Delta t\ v_{x_i}(t+1)$ {and similar for $y_i(t+1)$ and $z_i(t+1)$}

$time(t+1) = time(t) + \Delta t$