Simulation of the motion of N interacting particles in single plane (2D) under gravitational forces (N = 3)

PHYS 210
TERM PROJECT PROPOSAL
Karolina Bae (39091111)

Overview

- Motion of n-body particles can be approximated by gravitational forces between them.
- N-body problems may range from celestial bodies such as earth and moon to gas particles in a cloud.

Project Goals

- To write Matlab code to solve the n-body problem where n = 3.
- To correct any possible errors by conducting multiple tests.
- To visualize this 3-body motion using a visualizing software.

Mathematical Formulation

- Gravitational force between two particles is

$$F_G = \frac{G m_1 m_2}{r^2}$$

where G = Gravitational constant

m = mass of the particle

r = distance between two particles

- Total force can be found by superimposing forces exerted by every particle.

$$\sum_{i \neq n}^{N} \frac{G m_n m_i}{r_{ni}^2} \hat{r}_{ni}.$$

Numerical Approach

- N-body problem is solved by finite difference approximations (FDA) to calculate the particle's position and velocity, depending on its initial position and velocity.

- Using FDA, the particle's force and acceleration can be found.

Grid points: $X_j = x_{min} + (j-1)\,\Delta x$

Grid function notation : $f_j = f(x_j)$

$\rightarrow f'(x_j) \equiv f'(x_j) \approx f_{j+1} - f_j / \Delta x$

Visualization

- Matlab plotting function will be used to visualize the N-body system

Testing & Numerical Experiments

- Three particles will be given non-zero (different) mass.
- For simplicity, their initial velocities will be zero.
- Each particle’s various positions will be determined based on their initial position as time goes.

Project Timeline

Dates	Activities
10/15 – 10/27	Basic research, derive equations & begin code design
10/28 – 11/17	Implement code
11/18 – 11/21	Test code
11/22 – 11/28	Run numerical experiments, analyze data, begin report
11/29 – 12/01	Finish report
12/02	Project submission

References

- http://physics.princeton.edu/~fpretori/Nbody/intro.htm
- http://www.rsmas.miami.edu/personal/miskandarani/Courses/MSC321/lectfiniteDifference.pdf

Equilibrium Configuration of N-Identical charges

Nicole

Overview

- like charges repel and spread to maximize distance between each other
- charges interact by Coulomb's Law
- Once equilibrium is obtained, charges will be stationary
- Distributed symmetrically

Equations

$$F = \frac{qq}{4\pi\varepsilon_0 r} = ma$$

$$F_{drag} = -\gamma v$$

Where γ = drag coefficient

Goals/ Timeline

- Program and simulate n-charged particles on the surface of a sphere
- Determine shapes that correspond to equilibrium

- October: research, derive equations
- November
  1st half: program breakdown + start coding
  2nd half: write up + touch ups
  → hand in by end of November

Diffusion Limited Aggregation

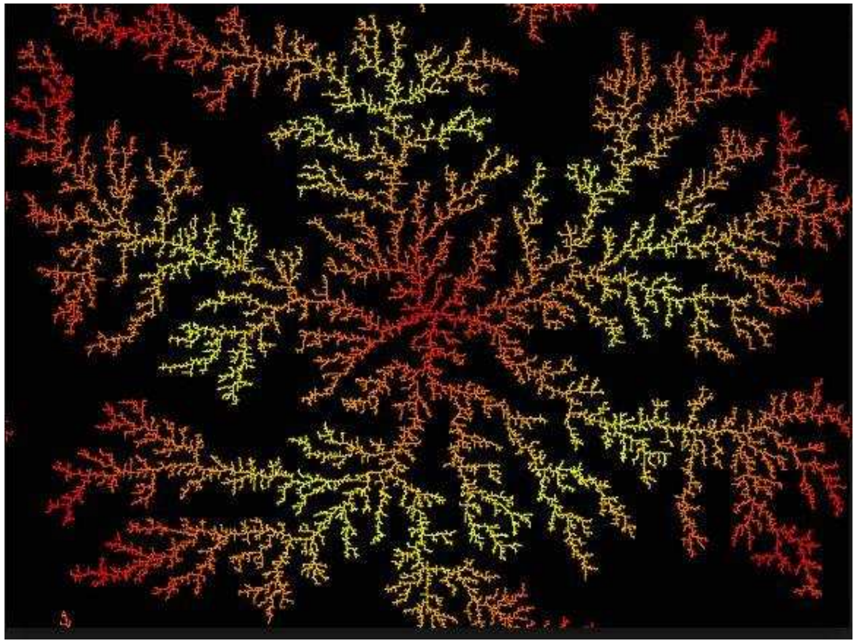
PHYS 210

Project Proposal

Jordan Coblin

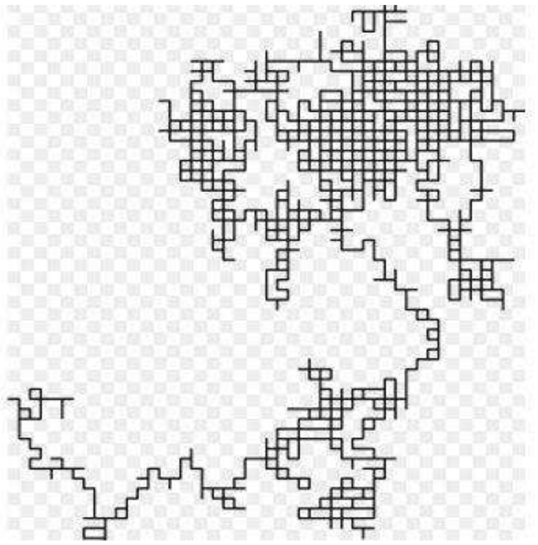
Overview

What is DLA?



- Formation of a fractal –like structure , called a "Brownian Tree" through the accumulation of particles that undergo movement as characterized by a random walk.

Overview



- **Random Walk :** used to describe random movement comprising of discrete steps.
- E.g. each step you take has an equal probability of being to the North, South, East or West. After 200 steps, what would your path look like?
- Collection of random variables to describe size/direction of each step (coin toss for 1D).

Real-World Examples

- Lighting Bolts (dielectric breakdown)
- Snowflake Formation
- Electrodeposition

Project Goals

- Produce a (2D/3D) DLA model using code written in MATLAB.
- Determine the correctness of the model through appropriate tests and comparison to known solutions.
- Investigate the effects of varying initial conditions (lattice dimension, seed position, sticking coefficient etc.).
- Analyze the fractal aspect of the model.

Mathematical Formulation

- Diffusion Equation

$$\frac{\delta \Phi(x, t)}{\delta t} = D \nabla^2 \Phi(x, t)$$

- Where:

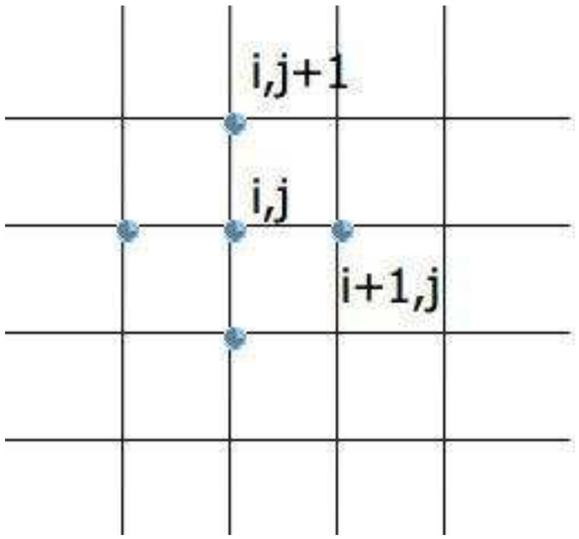
  - D is the diffusion coefficient (constant)
  - $\Phi(x,t)$ is the density at location x and time t

Mathematical Formulation

- Diffusion in a lattice

$$\frac{\delta\Phi}{\delta x} = \frac{\Phi_{i+1j,t} - \Phi_{i,j,t}}{a}$$

$$\frac{\delta^2\Phi}{\delta x^2} = \frac{\Phi_{i+1j,t} - 2\Phi_{i,j,t} + \Phi_{i-1,j,t}}{a^2}$$



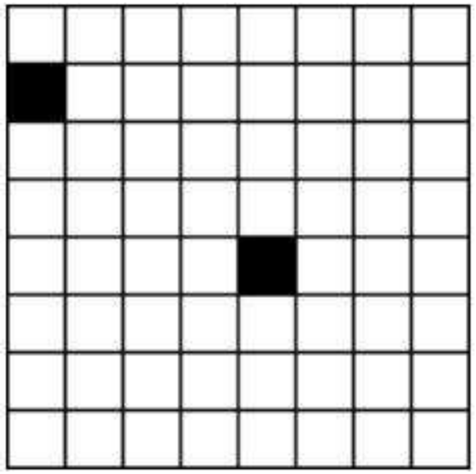
Where a is the lattice spacing

Numerical Approach

- Add a seed particle at the centre of the lattice (stationary).
- The next particle is added at a random position on a ring with radius R_{max} + BR, and centered at the seed.
- R_{max} is the radius of the cluster (changes with time).
- BR is the birth radius- an offset to create a new particle at a certain distance away from the cluster.

Numerical Approach

- Added particle then undergoes a random walk, either getting close enough to stick to the seed particle, otherwise exiting the lattice.
- Repeat particle addition n times.



Testing and Numerical Experiments

- Increase number of particles, beginning with a small number.
- Vary initial conditions to observe effects on aggregate formation.
- Introduce a sticking coefficient (probability of particle attaching to cluster).
- Determine the fractal dimension…

Testing and Numerical Experiments: Fractal

- Fractals have self-similarity on different scales – zoom in and the structure still looks the same.
- Fractional Dimension between 1 and 2 (between a line and a plane).

$$N = R_{max}^{D} \rightarrow D = \ln(N)/\ln(R_{max})$$

- Where N is the number of particles in the cluster, and D is the fractal dimension.

Project Timeline

October 22-31	Research, derive equations and begin code design.
November 1-15	Implement Code.
November 16-19	Test Code.
November 20-26	Run numerical experiments, perform analysis, begin report.
November 27-30	Finish Report.
November 30 – December 2	Submit Project.

References

- http://en.wikipedia.org/wiki/Diffusion-limited_aggregation
- http://pauli.uni-muenster.de/tp/fileadmin/lehre/NumMethoden/WS0910/ScriptPDE/Heat.pdf
- http://www.phas.ubc.ca/~berciu/TEACHING/PHYS349/DLA.pdf
- http://arxiv.org/ftp/arxiv/papers/1105/1105.5558.pdf
- https://www.philipithomas.com/files/diffusion.pdf

PHYSICS 210
TERM PROJECT PROPOSAL

Adam Dvorak

N-Body Problem

Goal: Simulate N bodies interacting due to gravitational force in three dimensions with an octave/matlab code

METHOD

-Force due to gravity between two bodies

$$F = -G(m_i)(m_j)(r_i - R_j)/(|r_i - R_j|^{3)}$$

-Newton's Second Law

$$F_i = m_i(a_i)$$

-Use finite difference approximation

$$a = F/m = f'(v_0) = (f(v_0 + \triangle t) + f(v_0)) / \triangle t$$

-Make $\triangle$t increasingly small to generate a smoother, more accurate simulation

-Supply initial conditions; position, velocity

Following Creation of the Simulation:

-Experiment with different initial conditions

-Study systems of two bodies orbiting one another

-Study the interaction of a third body colliding with the aforementioned system

Testing

-Compare results with current knowledge about interaction of N bodies due to gravity

-Use convergence testing to determine the order of the finite difference approximation $O(h^1)$, $O(h^2)$

Timeline

- Figure out all the approximations for the relevant equations by November 1st
-Write/implement code by November 15th
-Test code and run experiment by November 20th
-Analyze simulation, write report by November 25th
-Final details and submit by the end of November

Sources

http://en.wikipedia.org/wiki/N-body_problem

http://www.scholarpedia.org/article/N-body_simulations_(gravitational)

http://www.maplesoft.com/support/help/Maple/view.aspx?path=MathApps%2FNBodyProblem

A Simple Approach to Simulating Neural Networks

PHYS 210 Term Project Proposal

Farhad Fana

Overview

- A neural network is a system of individual constituent parts, neurons
- A neuron is made up of the main body (soma) and dendrites and axons.
- The dendrites and axons connect one neuron to another and communication happens through electrical pulses along these, this is the process of a neuron “firing”
- A neural network can be modeled as memory using an Ising spin model (up/down spin, on/off, firing/not firing)

Project Goals

- To write a MATLAB code which will simulate the memory of a neural network using the Ising model
- Use the Monte Carlo method to “test” the memory model and continue altering the states of the neurons
- Apply the technique to different patterns and test

Mathematical Formulation

- Energy function of the neurons in the Ising model:

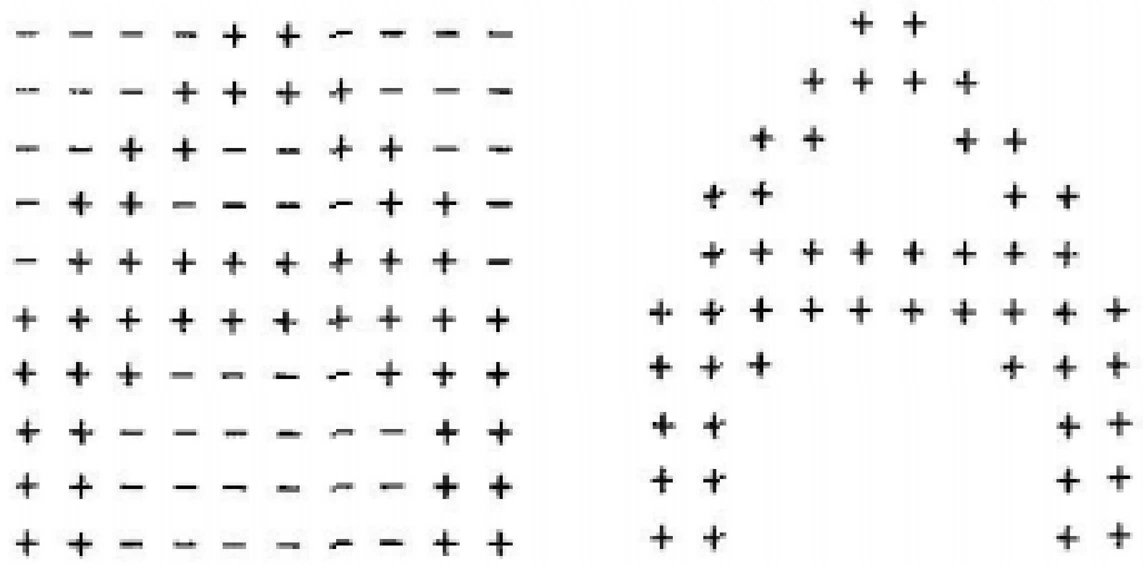
$$E = -\sum_{i,j} J_{(i,j)} s_i s_j \qquad (1)$$

- Energy stored in many patterns:

$$J_{(i,j)} = 1/M \sum_{m} s_i(m) s_j(m) \qquad (2)$$

Numerical Approach

- The pattern to be recalled is modeled as an Ising lattice:



- Where there is a distinct energy E associate with each pattern being recalled, related by equation (2). In the case of one pattern, 1/M=1.

Numerical Approach

- The Monte Carlo technique applies an algorithm which evaluates the spin value of each neuron and compares it to the calculated energy associated with the pattern.
- With each iteration, this technique will alter (or leave unchanged) the neurons' state until the most stable state is achieved, and the pattern recalled.

Testing

- Test with various parameters:
  - Different patterns
  - Total number of patterns stored
  - Vary amount of randomness in patterns
  - Many other variations...

Timeline

Dates	**Objectives**
Oct 20th – Oct 27th	Research, begin formulation and code planning
Oct 28th – Nov 2nd	Implement code
Nov 3rd – Nov 15th	Test code
Nov 16th – Nov 25th	Numerical experimentation and data analysis
Nov 26th – Dec 2nd	Finish report and submit project

References

- N.J. Giordano and H. Nakanishi, *Computational Physics, 2nd Edtion*, Prentice Hall, West Lafayette, 2005
- http://en.wikipedia.org/wiki/Ising_model

N-Body Gravitational Interaction using Finite Difference Approximation

Raphael Goemans

Phys 210 Project Presentation

Overview

- The User will specify the number N of particles at the beginning of each simulation.
- It is then possible to calculate the gravitational force exerted by any two particles on one another.

Project Goals

- Write a MATLAB code which will perform the following:
  - Can be given many different initial conditions.
  - The code will then be able to predict the motion of each particle involved.

Mathematical Formulation

- The equation for Newton’s law of universal gravitation:

$$F = G\frac{m_1 m_2}{r^2}$$

Mathematical Formulation

- By combining Newton’s second law, as well as his law of gravitation, we can find the basic equation of motion in vector form:

$$m_i \, \mathbf{a}_i = G \sum_{j=1,\, j\neq i}^{N} \frac{m_i \, m_j}{r_{ij}^2} \, \hat{\mathbf{r}}_{ij} \; , \qquad i = 1, 2, \, \ldots \, N$$

rij is the magnitude of separation between two particles i and j.

Testing and Numerical Experiments

- I will test this code using simple initial conditions at first (N = 1, 2, or 3) with all particles having equal mass.
- After this initial testing process is complete It can be tested with many more arbitrary conditions affecting the number of particles N, and the mass of these particles.

Testing and Numerical Experiments

- I will also look at the general movement of the particles to make sure there isn’t any particle jump at any time or other weird problems.

Testing and Numerical Experiments

- I will use the process known as discretization:
  - The unknown function will be defined on some interval $0 \leq t \leq t_{\max}$
  - Therefore, the unknown function will be defined at a finite number of times t^n, n=1,2,3…n.
  - This discretization process is necessary in order to solve this problem because computers can't make infinite calculations, and we need to change the differential equations into algebraic equations.

Project Timeline

10/22/13	Project Presentation.
10/22/13 – 10/29/13	Research and Initial design of code.
10/29/13 – 11/10/13	Code Write-up.
11/10/13 – 11/12/13	Code Debug.
11/12/13 – 11/14/13	Code testing with varying initial conditions
11/14/13 – 11/20/13	Collection of Data with graphical simulations.
11/20/13 – 12/1/13	Finish the write-up of the project.
12/1/13 – 12/2/13	Pray everything is done and working as it should.
12/2/13	Project Due Date.

References

- http://en.wikipedia.org/wiki/Finite_difference
- http://en.wikipedia.org/wiki/Newton's_law_of_universal_gravitation
- http://en.wikipedia.org/wiki/N-body_problem

Simulation of N-Particles in R^3 Space

PHYS 210 Term Project Proposal

Keith Grant

Overview

- N-Body problem shows the individual motion and forces acting on a group of objects gravitationally
- Can be used to help understand the dynamics of globular clusters (ei – collisions)
- This simulation will be a more simple model in xvs– will negate the ecliptic nature of celestial bodies
- Will also negate general relativity, dark matter, and object collision effects.
- Will be total elastic collisions.

Project Goals

- To write a a code in MATLAB which simulates the gravitational effects on N number of particles in three dimensional space over an allotted time
- To produce consistent results in various initial conditions including various combinations of the particle's initial velocity, and position.
- To present these interactions graphically using a simulator
- To ensure all major physical laws are followed (ei - Conservation of momentum & energy).

Mathematical Formulation

Gravitational Interaction between two particles:

$$\vec{F}_g = G\frac{m_1 m_2}{r^2}\hat{r} \Rightarrow G\frac{m_i m_j}{|\vec{r}_j - \vec{r}_i|^2}\frac{(\vec{r}_j - \vec{r}_i)}{|\vec{r}_j - \vec{r}_i|}$$

Newton's Second Law

[redacted]

$$\vec{F} = m\vec{a} = \frac{d^2\vec{x}}{dt^2}$$

Giving the relation: $\displaystyle \frac{d^2\vec{x}_i}{dt^2} = G\sum_{j=1}^{n}\frac{m_j}{|\vec{r}_j - \vec{r}_i|^2}\frac{(\vec{r}_j - \vec{r}_i)}{|\vec{r}_j - \vec{r}_i|}, i \neq j$

Numerical Approach

Taylor series will be used to solve for position and velocity at time ‘t’ in small steps of t

h=step size

$$f(t_0+h)=f(t_0)+hf'(t_0)+\frac{1}{2}h^2y''(t_0)+O(h^3)$$

Can then use the this to take the differential to find position at a later time ‘t’

$$\vec{x}_i(t+h)=\vec{x}_i(t)+h\frac{d\vec{x}_i}{dt}(t)+\frac{h^2}{2}\frac{d^2\vec{x}_i}{dt^2}(t)$$

Should be noted that there is an significant error with this lower order.

Testing & Numerical Experiments

- Testing
  - Using the laws of conservation of momentum and energy are satisfied (shows code is running smoothly)
  - Also check Δx to ensure no abnormal / unexpected changes in position (design flaw)
- Numerical Experiments
  - Use various initial states of particles as well as the number of N particles.

Project Timeline

Date	Activities
October 22 – October 28	Basic research of of N-body problem and mathematics, begin code design
October 29 – November 6	Write & Implement Code
November 7 – November 10	Test Code
November 11 – November 15	Run Numerical Experiments, Begin Report
November 16 – November 20	Analyize Data, Begin Presentation, Continue Report
November 21 – November	Finish Presentation & Report
November	Present Project
November 29	Submit Project

References

- http://en.wikipedia.org/wiki/N-body_problem
- http://bh0.phas.ubc.ca/210/Doc/term-projects/kdv.pdf
- http://vuduc.org/teaching/cse8803-pna-sp08/slides/cse8803-pna-sp08-20.pdf

Simulation of a Simple Neural Network

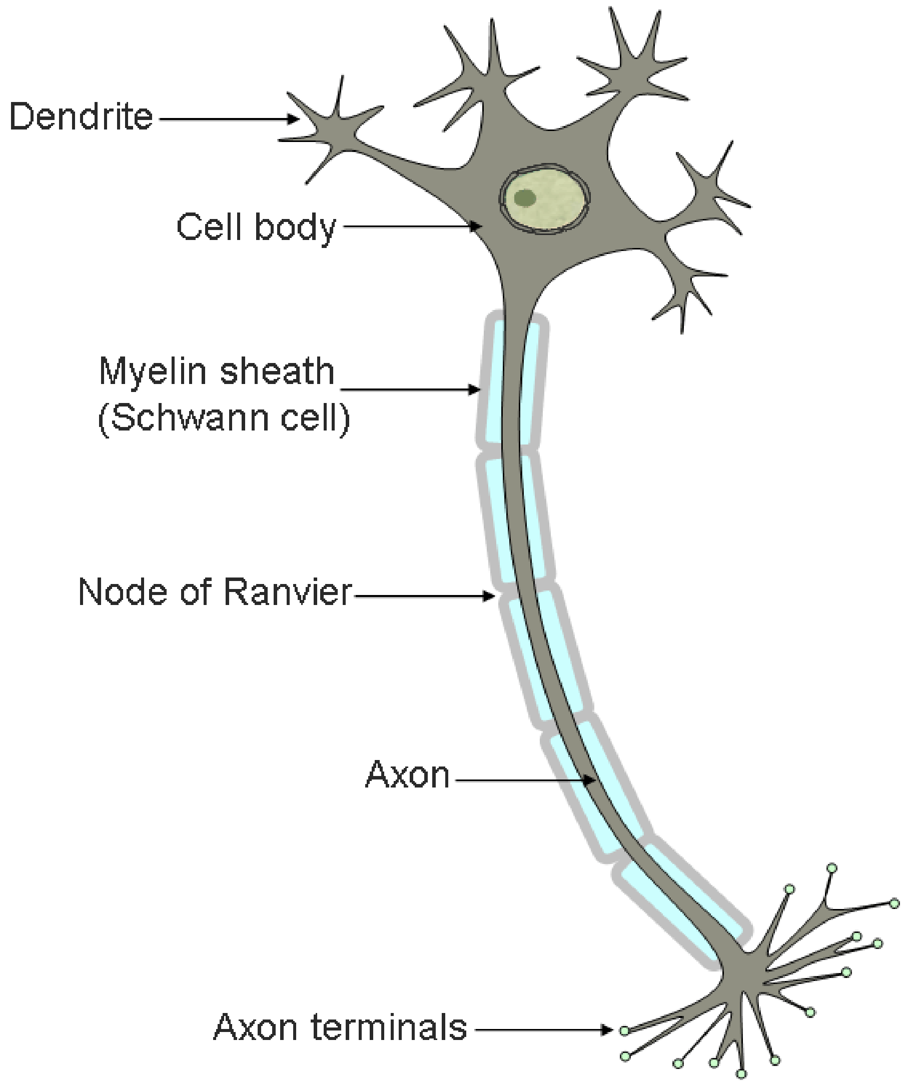
Phys 210 Term Project Proposal

Kevin Guo

33738121

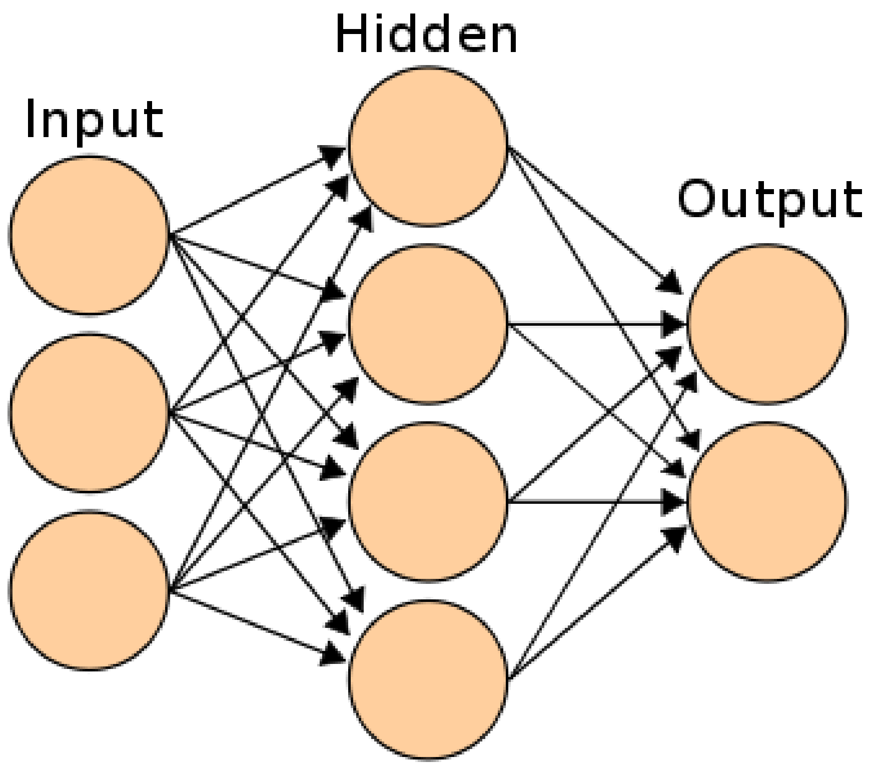
Overview

- The human brain consists of about 10^{12} neurons
- They are electrically active thus can communicate to other neurons carried by dendrites and axons
- Neurons generally have many dendrites and accept many inputs from other neurons



Project Goals

- To write a MATLAB (octave) code that uses an Ising model to simulate a simple neural network
- Model each neuron as a simple Ising spin (Means they can be firing or not firing)
- Using a step function to approximated firing and not firing instead of using a curve
- Include the spin's effect on its neighbours to highly mimic the interconnected nature of a real neural network



Mathematical Formulation

- Store a single pattern: (1)

$$J_{i,j} = \frac{1}{M} \sum_{m} s_i(m)\, s_j(m) \,,$$

- The most energetically stable pattern: (2)

$$E(m) = -\sum_{i,j} J_{i,j} s_i\, s_j = -\sum_{i,j} s_j(m)\, s_i(m)\, s_i\, s_j \,.$$

- Effective energy of neural network: (3)

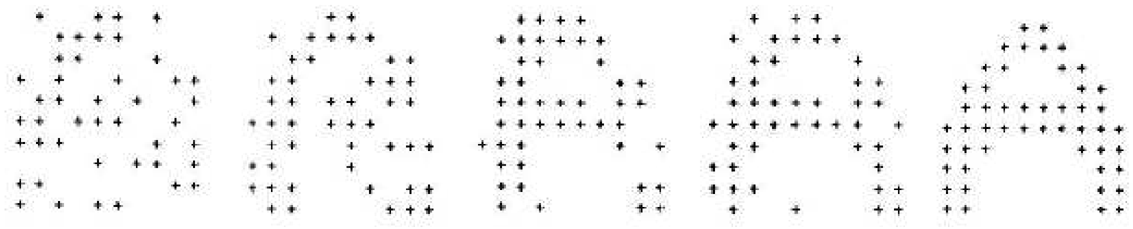
$$E = -\sum_{i,j} J_{i,j}\, s_i\, s_j \,,$$

Programming:

- Given row and column m,n the spin number is: (3)

$$i = N(\mathrm{m} - 1) + \mathrm{n} \,.$$

- **Testing and Numerical Experiments**
    - **Insure that the network fits the equations provided on the pervious page.**
    - **The equation (1) on the pervious page regarding storing a pattern can be altered to have a damage the pattern by setting elements of the matrix to 0 with a set probability**
- Ex: In decreasing probability



- **Project Timeline**

Dates	Activities
Week of Oct 21	Research into code & subject
Week of Oct 28 – Nov 11	Work on code
Week of 18	Test code
Week of 25	Finish Report
Dec 1	Review Project
Dec 2	Submit Project

- **References**

N. Giordano, *College Physics: Reasoning and Relationships*, Cengage Learning, Stramford , (2008)

ONE DIMENSIONAL TIME-DEPENDENT SCHRODINGER EQUATION

:: PHYS 210 TERM PROJECT PROPOSAL

:: ANDREW HAJDA

OVERVIEW

- The Schrödinger equation is the correct basis of nonrelativistic quantum mechanics.
- Solutions to Schrödinger's equation describe molecular, atomic, subatomic, macroscopic and (possibly) the whole universe.
- Solutions to Schrödinger's equation describe molecular, atomic, subatomic, macroscopic and (possibly) the whole universe.

PROJECT GOALS

- To use the Schrödinger equation to solve a one dimensional infinite potential well model.
- Will use Schrödinger equation as it applies to one particle moving in one dimension.

Mathematical Formulation

- $i\hbar\frac{\partial}{\partial t}\psi(x,t) = -\frac{\hbar^2}{2m}\frac{\partial^2}{\partial x^{\partial}}\psi(x,t) + V(x)\psi(x,t)$
- $\psi(x,t) = [Asin(kx) + Bcos(kx)]e^{-i\omega t}$ is the wave function of a free particle.

Numerical Approach

- Use Crank Nicholson Approximation to evaluate $\frac{\partial^2}{\partial x^\partial} \psi(x, t)$ at both time steps t_{n+1} and t_n
- $\frac{\partial}{\partial t} \psi\left(x_j, t_n\right) \cong \frac{\psi_j^{n+1} - \psi_j^n}{\Delta t}$
- $\frac{\partial^2}{\partial x^\partial} \psi(x, t) \cong \frac{\psi_{j+1}^{n+1} - 2\, \psi_j^{n+1} + \psi_{j+1}^{n+1}}{\Delta x^2} + \frac{\psi_{j+1}^{n} - 2\, \psi_j^{n} + \psi_{j-1}^{n}}{\Delta x^2}$

VISUALIZATION AND PLOTTING

- MATLAB
- SOMETHING FOR INTERACTIVE ANALYSIS AND MPEG ANIMATION.

Testing & Numerical Experiments

- Using fact that there is a conservation law for $\psi(x, t)$ (Probability Density function): $\int_0^1 \psi(x,t)\, \psi^*(x,t)dx$
- Check the behavior of simulations and results to known working models.
- Investigate different energy eigenstates of the wave function.
- Investigate different wavenumbers and there behavior.

Project Timeline

Date	Activities
10/20-10/31	Do research, derive equations
11/01 – 11/15	Implement code
11/16 – 11/20	Test code
11/21 – 11/27	Run numerical experiments, analyze data, begin report
11/28-12/02	Finish report
12/02	Submit Project

References

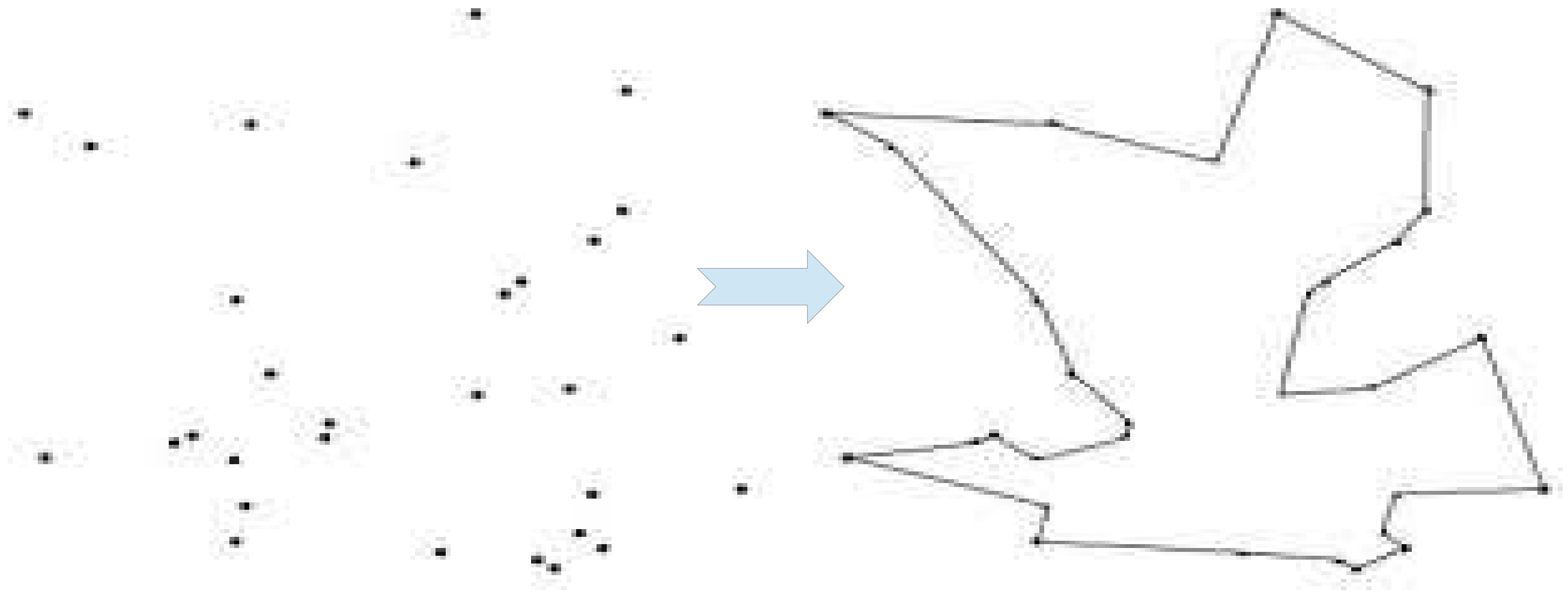
- http://en.wikipedia.org/wiki/Particle_in_a_box
- http://laplace.physics.ubc.ca/210/Doc/term/schrodinger.pdf
- http://web.cecs.pdx.edu/~gerry/class/ME448/notes/pdf/CN_slides.pdf

Questions?
Comments?
Suggestions?

Thank you !

Solutions to the 'Travelling Salesman Problem' with a Genetic Algorithm

- Overview
  - The 'Travelling Salesman Problem' (TSP) is the problem of having an arbitrary number of points to visit, and figuring out the 'optimal' (minimal distance/time) to visit all the points and return to the starting point.



- The solution to the problem is the list of points to visit in the order to visit them

The problem with computing the solution is that the 'straight forward' algorithm of computing the distance of every permutation of the N sized set of points is proportional to N-1 factorial.

For a set of 31 points like in the problem before, there are 2.65x10^32 unique lists to evaluate and compare to solve the problem.

A Genetic Algorithm works analogously to natural selection.

A population of initial lists are used to generated a new population of lists that are better solutions to the problem than the first list. This process is repeated until ideal or satisfactory solutions are found.

The lists are processed every cycle: the worst lists may be eliminated, and replaced with copies of the best lists, random mutations change the lists, and lists combine.

- Project Goals
  - To use MATLAB to write a genetic algorithm that can solve the TSP for an arbitrary number of points
  - To determine how the number of iterations of the algorithm is proportional to the size of the problem
  - To investigate how changing mutation rates, population size, etc. affect how quickly and reliably a solution is produced.

- Approach
  - Representation of Problem set

    The set of points to visit stored in a Nx2 matrix
  - Representation of solutions

    Each solution is a permutation of the vector

    [1,2,3...,N-1], where the numbers represent the index of the point travelled to.
  - Fitness function

    The fitness of a path is the total distance travelled.

    Paths travelling the least distance are the most fit.
  - Mutation/Crossover

    Randomly switching elements in the solution, swapping sections of two solutions.

- Testing

  Sets of points with obvious solutions (circle of points) can be put through the algorithm to see if it produces the ideal solution.

- Visualization and Plotting Tools

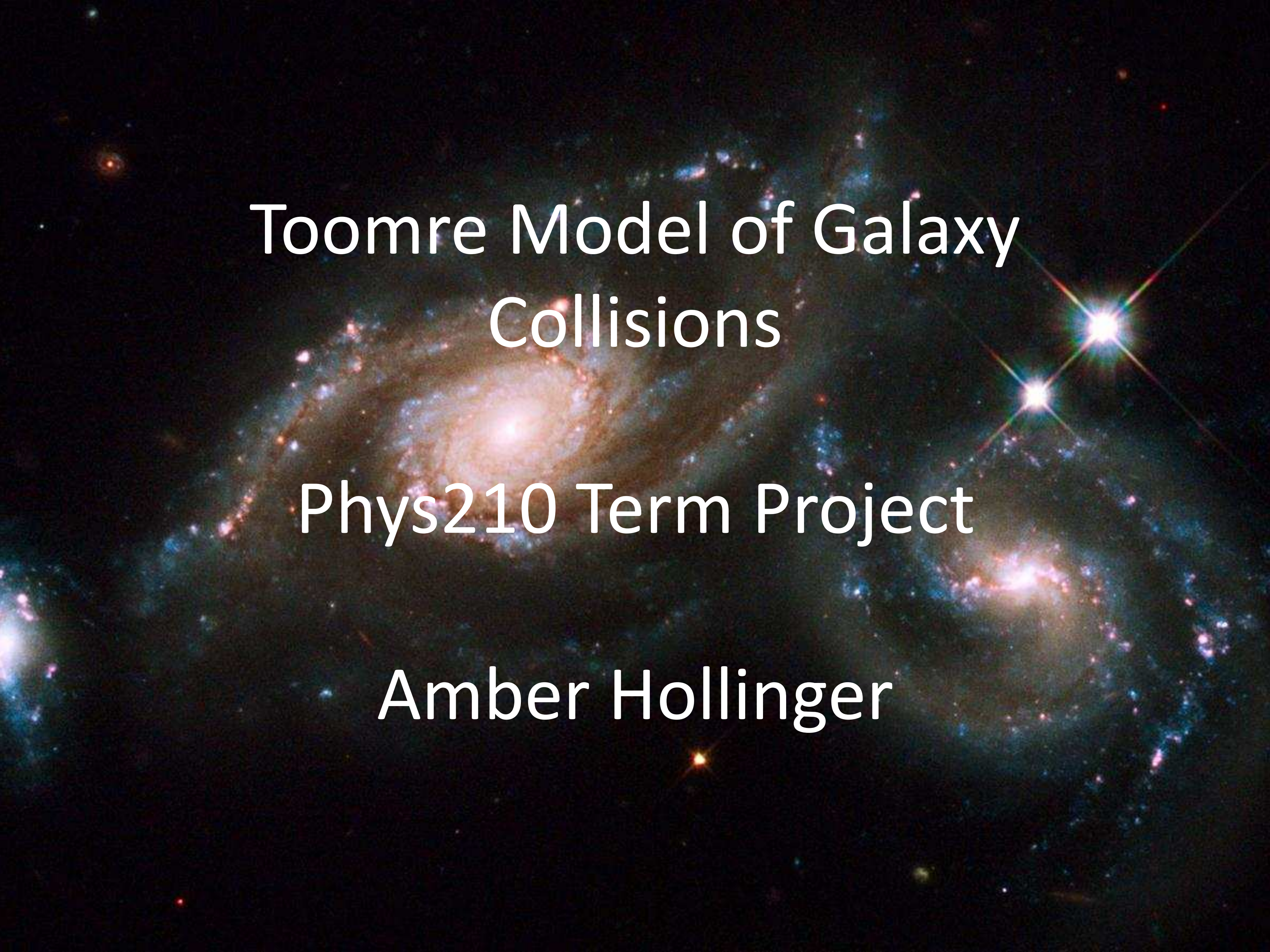
  I'll use MATLAB facilities to plot data

- Project Timeline

  Oct.18-Oct.25: Plan design of Genetic Algorithm
  Oct.26-Nov.2: Implement code
  Nov.3-Nov.10: Research and implement code to iterate for large set of solutions
  Nov.11-Nov.18: Analyze data, start report
  Nov.18-Nov.25: Finish and submit report

- References
  - Solving the Assignment problem using Genetic Algorithm... http://www.iaeng.org/IJAM/issues_v36/issue_1/IJAM_36
  - Optimal Population Size and the Genetic Algorithm

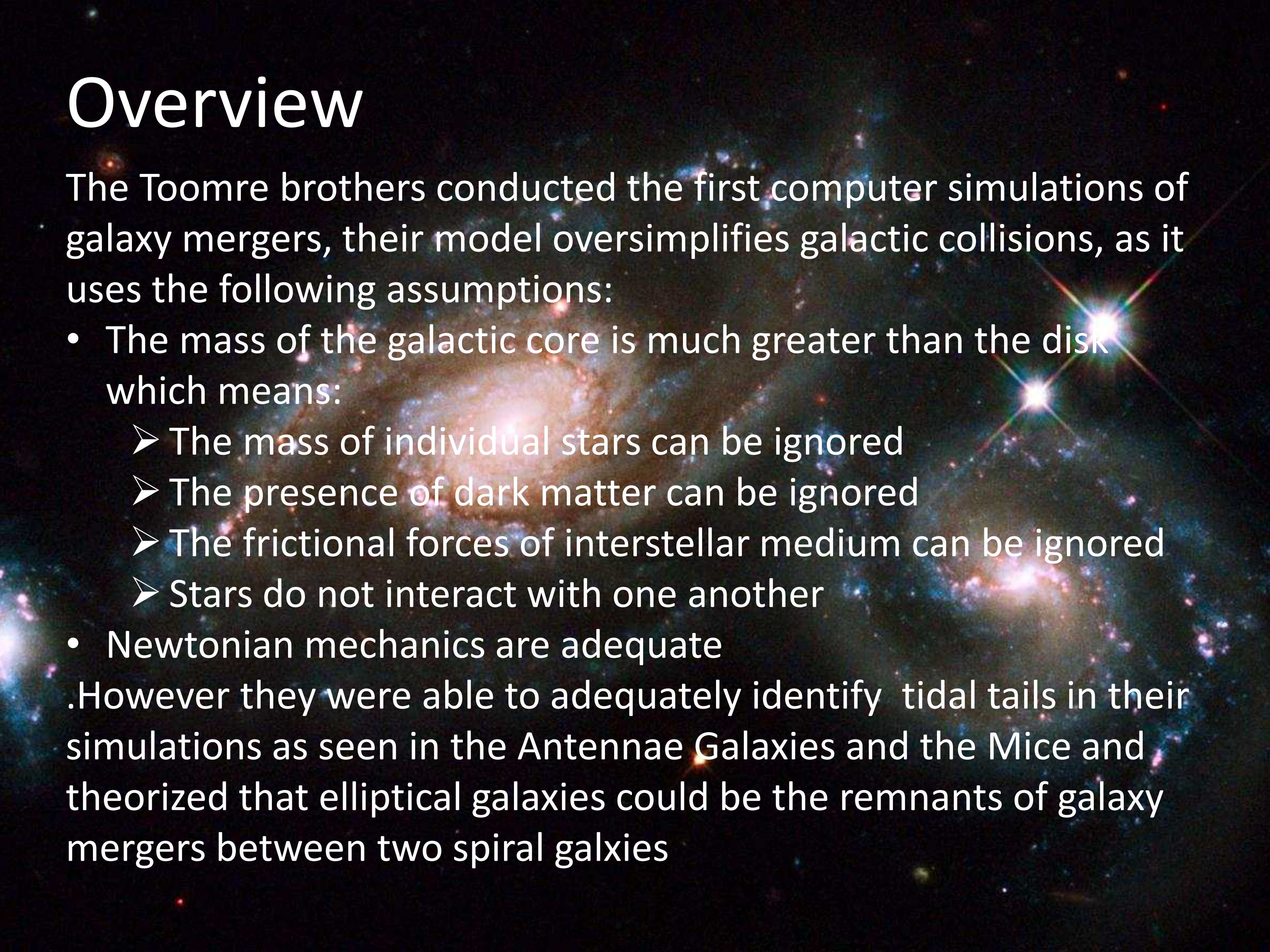
    http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1
  -



Toomre Model of Galaxy Collisions

Phys210 Term Project

Amber Hollinger



Overview

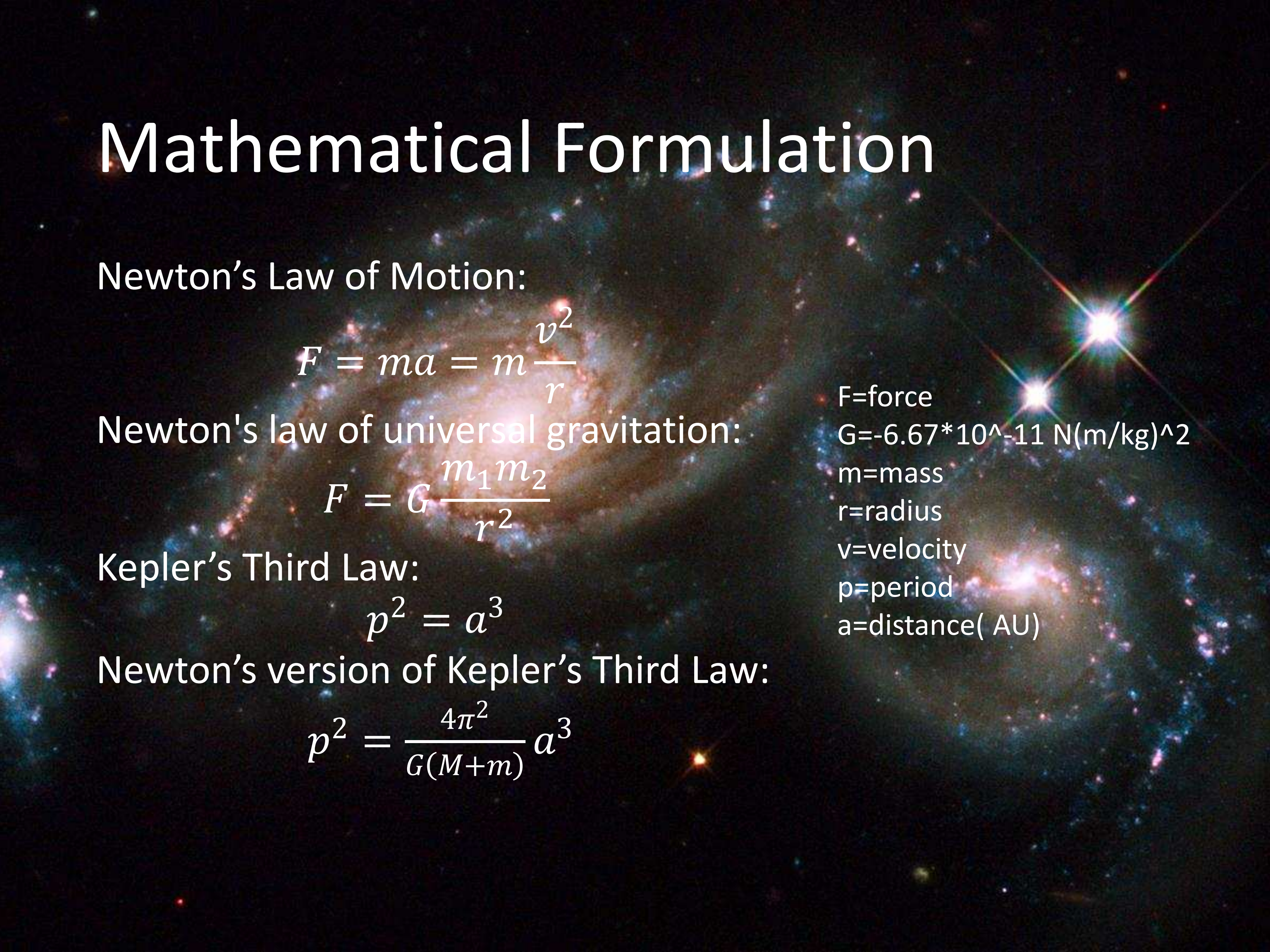
The Toomre brothers conducted the first computer simulations of galaxy mergers, their model oversimplifies galactic collisions, as it uses the following assumptions:

- The mass of the galactic core is much greater than the disk which means:
  - ➢ The mass of individual stars can be ignored
  - ➢ The presence of dark matter can be ignored
  - ➢ The frictional forces of interstellar medium can be ignored
  - ➢ Stars do not interact with one another
- Newtonian mechanics are adequate

.However they were able to adequately identify tidal tails in their simulations as seen in the Antennae Galaxies and the Mice and theorized that elliptical galaxies could be the remnants of galaxy mergers between two spiral galxies

Goals

1. Use octave to create an accurate visual simulation of galaxy collisions in 3D space.
2. To use various initial velocities, positions, angles of approach, masses of cores, and the number of stars
3. To be able to observe the effects of varying these initial conditions
4. To observe whether or not the collision of two elliptical galaxies will result in an elliptical galaxy
5. To ensure that the model is correct by checking the conservation of kinetic and potential energy



Mathematical Formulation

Newton’s Law of Motion:

$$F = ma = m\frac{v^2}{r}$$

Newton's law of universal gravitation:

$$F = G\frac{m_1 m_2}{r^2}$$

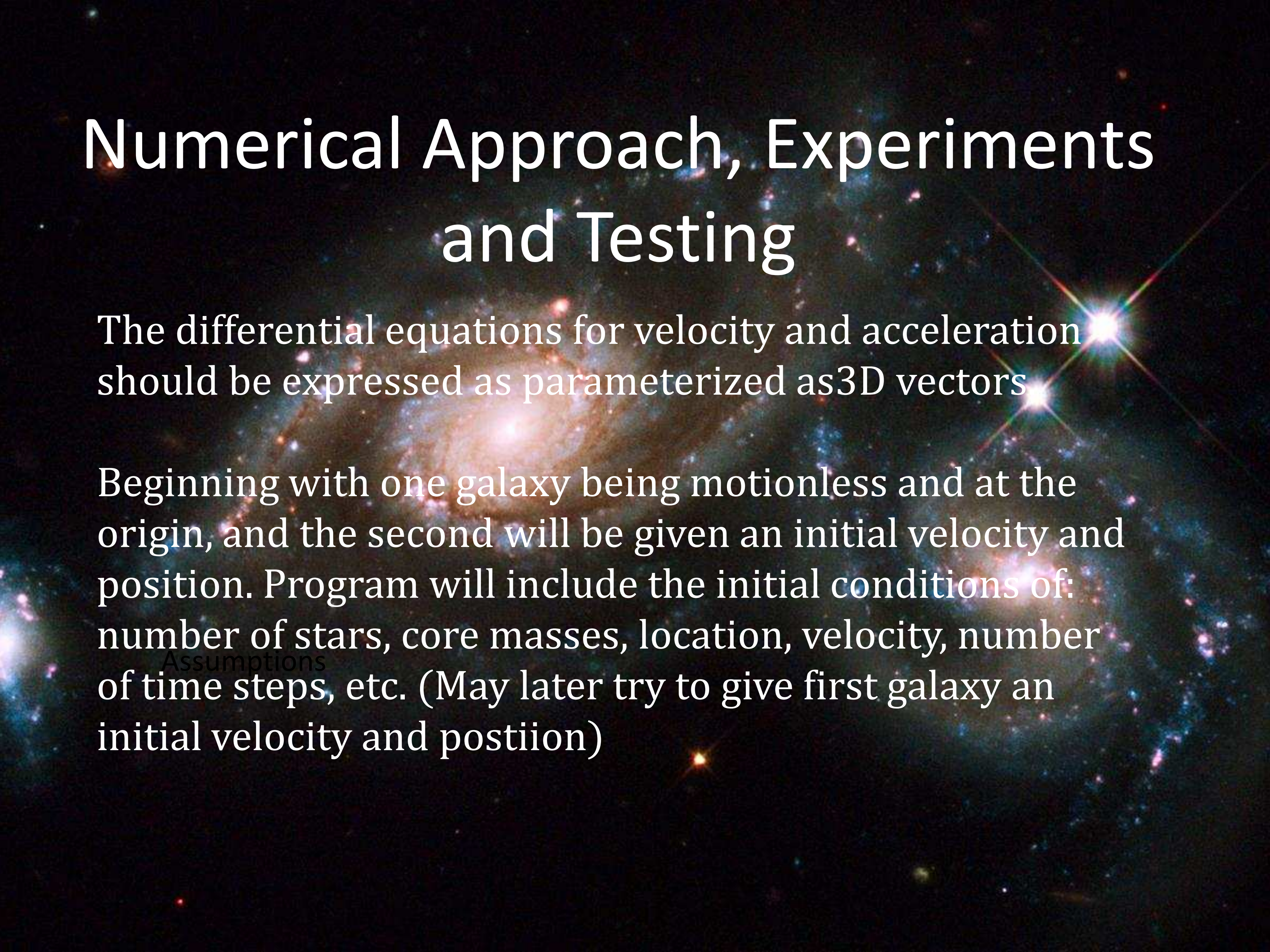
Kepler’s Third Law:

$$p^2 = a^3$$

Newton’s version of Kepler’s Third Law:

$$p^2 = \frac{4\pi^2}{G(M+m)}a^3$$

F=force
G=-6.67*10^-11 N(m/kg)^2
m=mass
r=radius
v=velocity
p=period
a=distance(AU)



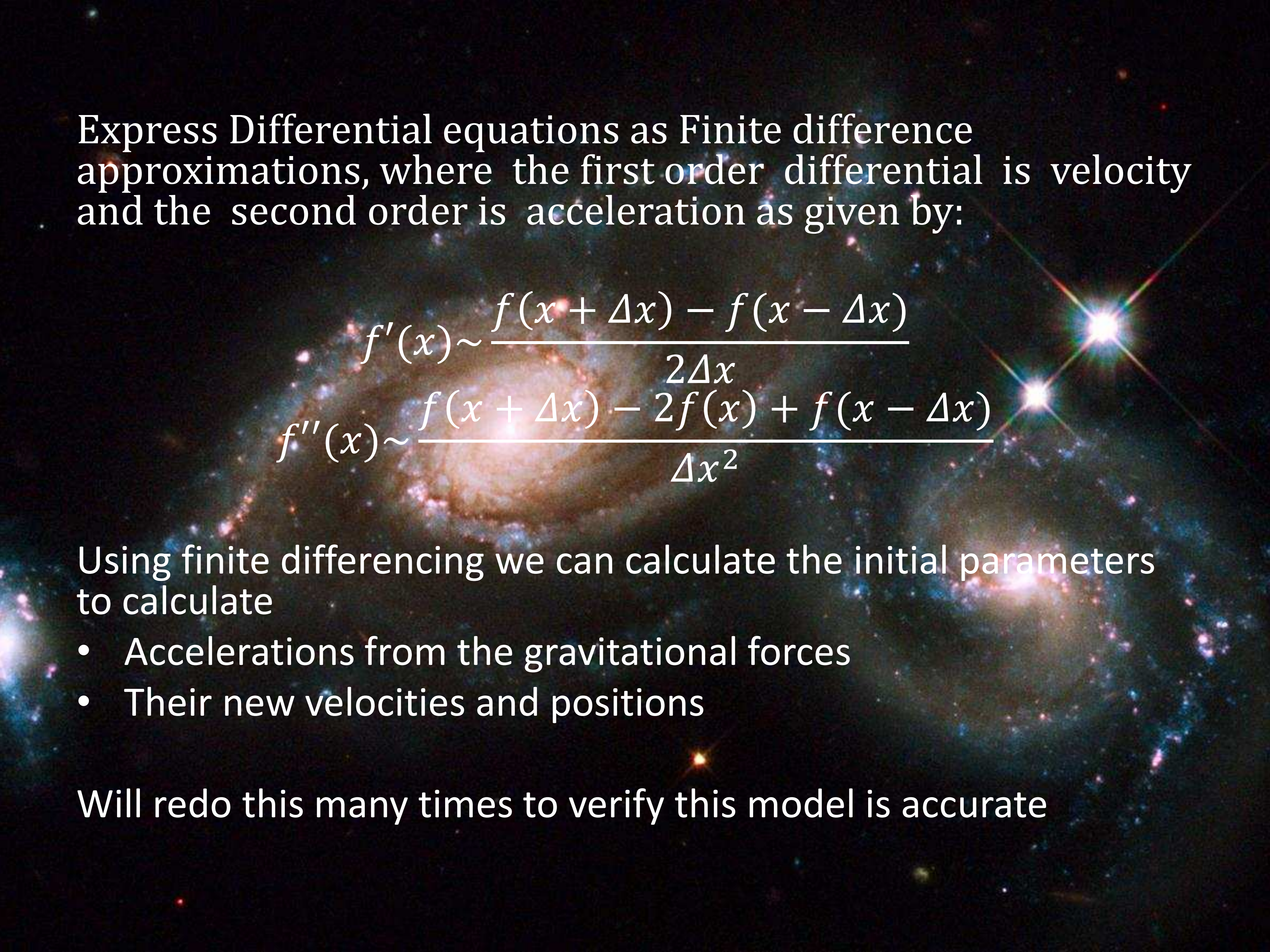
Numerical Approach, Experiments and Testing

The differential equations for velocity and acceleration should be expressed as parameterized as3D vectors

Beginning with one galaxy being motionless and at the origin, and the second will be given an initial velocity and position. Program will include the initial conditions of: number of stars, core masses, location, velocity, number

Assumptions

of time steps, etc. (May later try to give first galaxy an initial velocity and postiion)



Express Differential equations as Finite difference approximations, where the first order differential is velocity and the second order is acceleration as given by:

$$f'(x) \sim \frac{f(x + \Delta x) - f(x - \Delta x)}{2\Delta x}$$

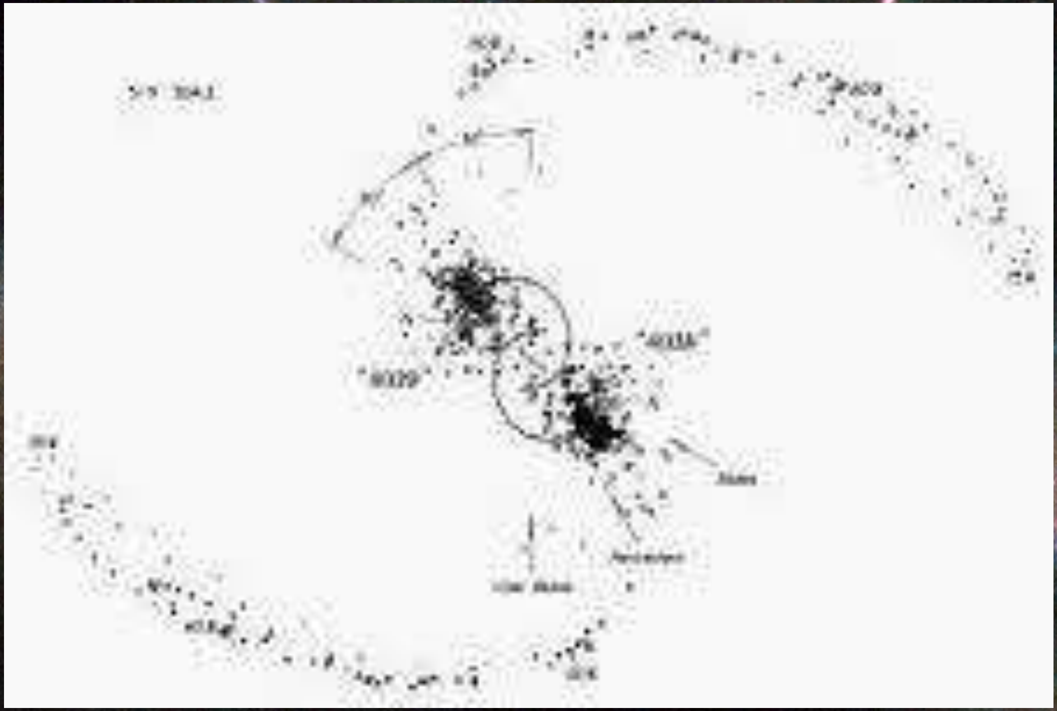
$$f''(x) \sim \frac{f(x + \Delta x) - 2f(x) + f(x - \Delta x)}{\Delta x^2}$$

Using finite differencing we can calculate the initial parameters to calculate

- Accelerations from the gravitational forces
- Their new velocities and positions

Will redo this many times to verify this model is accurate

I will compare my results with the Toomre model of the collision of the Antennae galaxies. By using the same initial parameters



I will attempt to recreate the collision of Andromeda and the Milky Way and campare it to current predictions

Timeline

Date	Goal
October 21-26	Begin researching and designing code
October 27-November 7	Implement Code
November 8-11	Test Code
November 12-25	Run numerical experiments, begin report and analyze data
November 26-28	Finish report
November 29	Submit report

References

References

http://en.wikipedia.org/wiki/Alar_Toomre
http://en.wikipedia.org/wiki/Galaxy_merger
http://kinotonik.net/mindcine/toomre
http://en.wikipedia.org/wiki/Antennae_Galaxies
http://cas.sdss.org/dr3/en/proj/basic/galaxies/collisions.asp

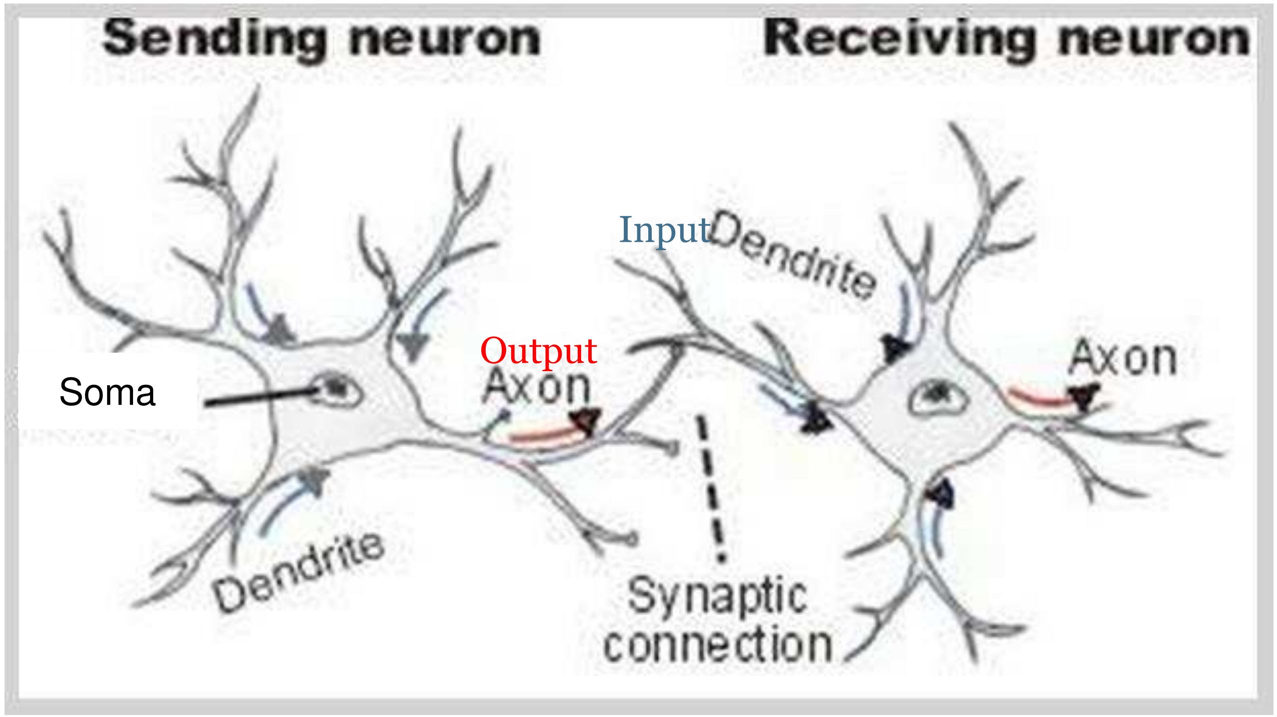
Simulating Content-Addressable Memory of a Neural Network

PHYS 210 – Introduction to Computational Physics

By: Jeffrey Hao-Chan Huang (29595113)

Structure of Neural Networks

- Basic Unit of a Neural Network = A Neuron



- Communication between neurons is achieved through electrical signals outputted by the axon of one neuron that is inputted into the dendrite of another neuron through the synapse.
- Because there are ~10^4 dendrites on a single neuron and the axon is split into ~10^4 branches, it accepts numerous inputs and outputs a signal to numerous other neurons.

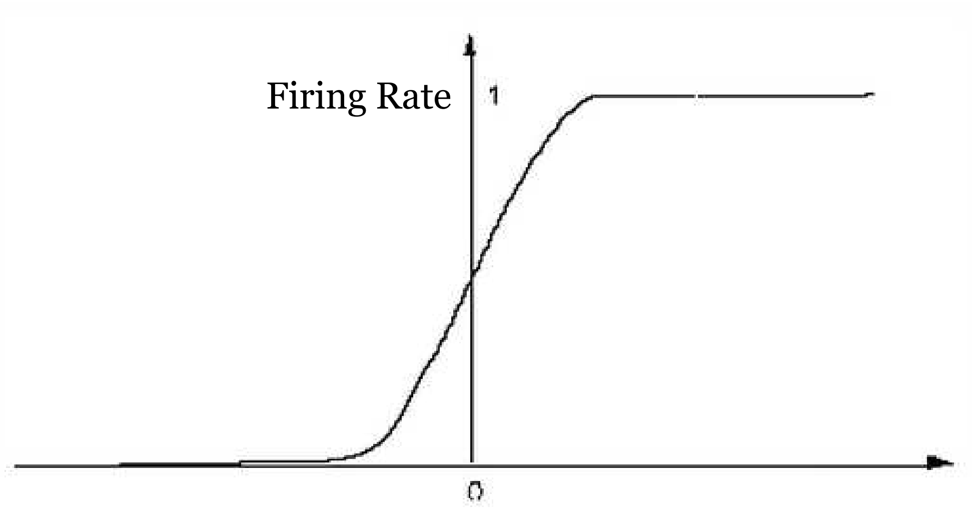
Firing Rate of a Neuron

- A neuron outputs a signal through its axon in a periodic manner.
- The rate at which the output signal is fired is a function of the input signals and the strength of the corresponding synaptic connection.

$$\boxed{\text{Inhibitory Synapse}} \longrightarrow \boxed{\text{Excitory Synapse}}$$

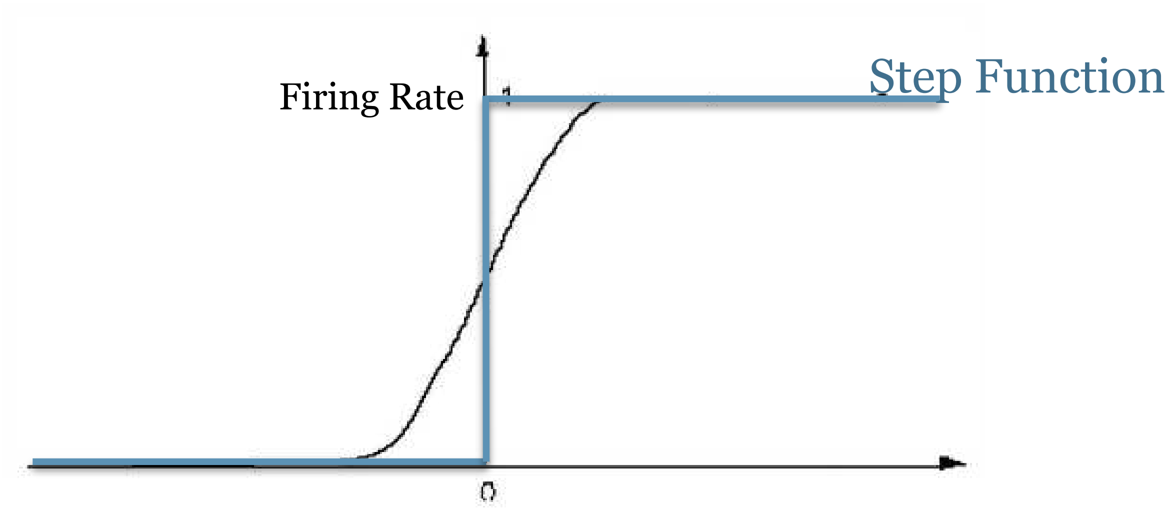
Inhibitory Synapse	Excitory Synapse
High Rate of Input ⇩ Low Rate of Output	High Rate of Input ⇩ High Rate of Output

- Range of a neurons output signal firing rate:



Modelling Firing Rate

- The firing rate of a neuron at any given time is modelled by a step function:

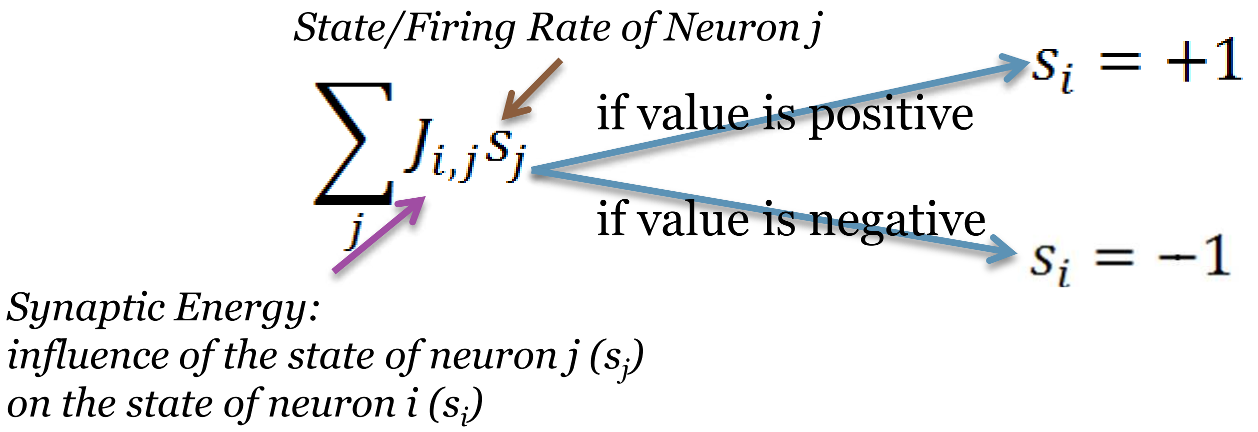


- As a result of this approximation, a neuron can be modelled as a simple Ising spin with 2 possible states:

State	**S=+1**	**S=-1**
Firing Rate	1	0

Modelling Firing Rate

- The state of neuron i (its output signal firing rate) is determined by the superposition of the effective fields established by its interactions with other neurons , which provide its input signals.


State/Firing Rate of Neuron j
$\sum_j J_{i,j} s_j$
if value is positive
$s_i = +1$
if value is negative
$s_i = -1$
Synaptic Energy:
influence of the state of neuron j (s_j)
on the state of neuron i (s_i)

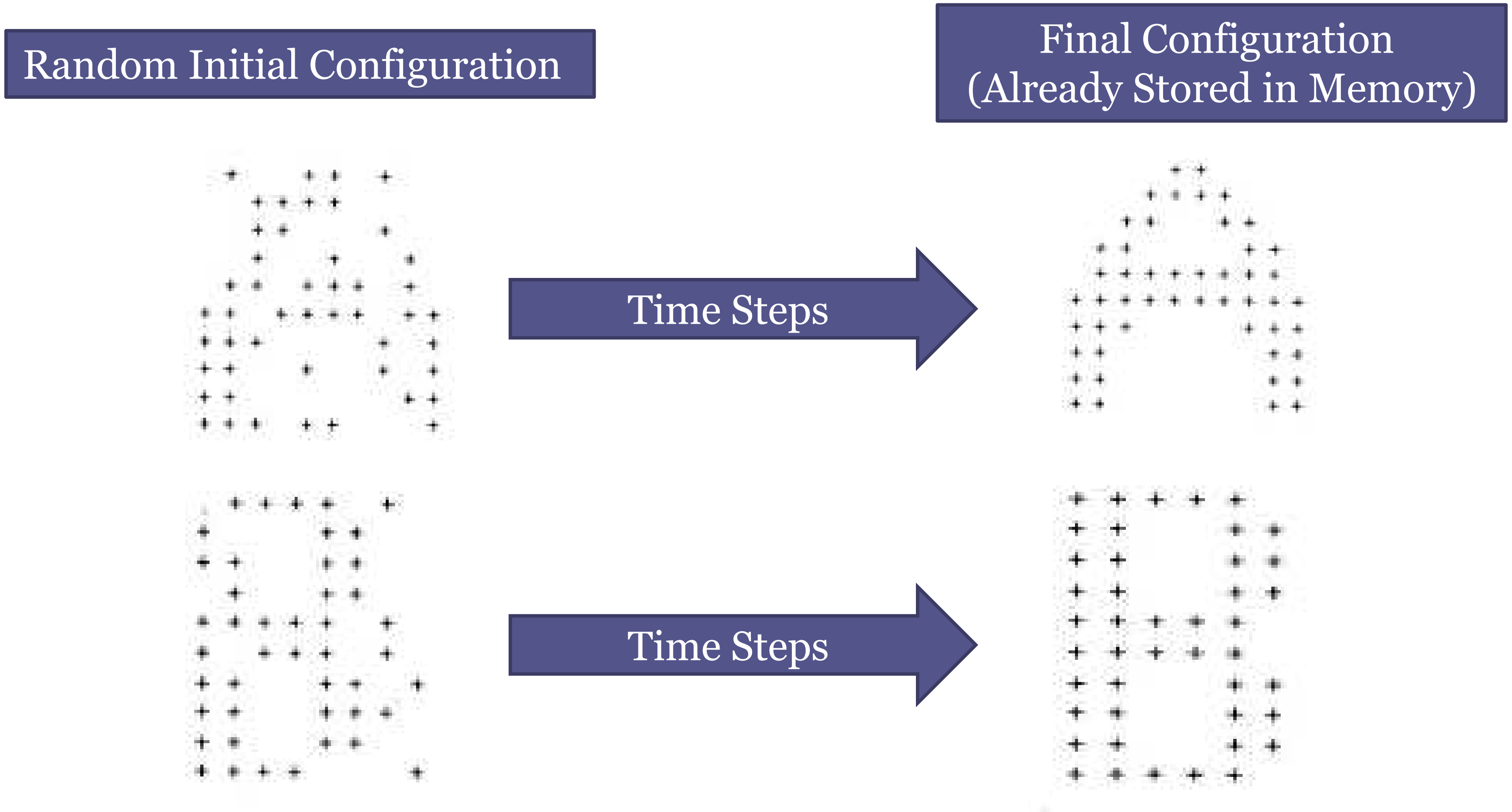
- Effective energy of a neural network where each neuron i is connected to all other neurons in the network:

$$E_{network} = -\sum_{i,j} J_{i,j} S_i S_j$$

- The neural network prefers states that minimize this energy.

Term Project Objective

- Design code using MatLab to simulate a 100 x 100 lattice of interconnected neurons that is capable of content-addressable memory and recognition.


Random Initial Configuration
Final Configuration (Already Stored in Memory)
Time Steps
Time Steps

Mathematical Formulation

- Assuming:
  - Each neuron in the lattice has synaptic connections to all the other neurons.
  - The connection between any 2 neurons is symmetric.

  Effective Energy of a Neural Network:

$$E_{network} = -\sum_{i,j} J_{i,j} S_i S_j$$

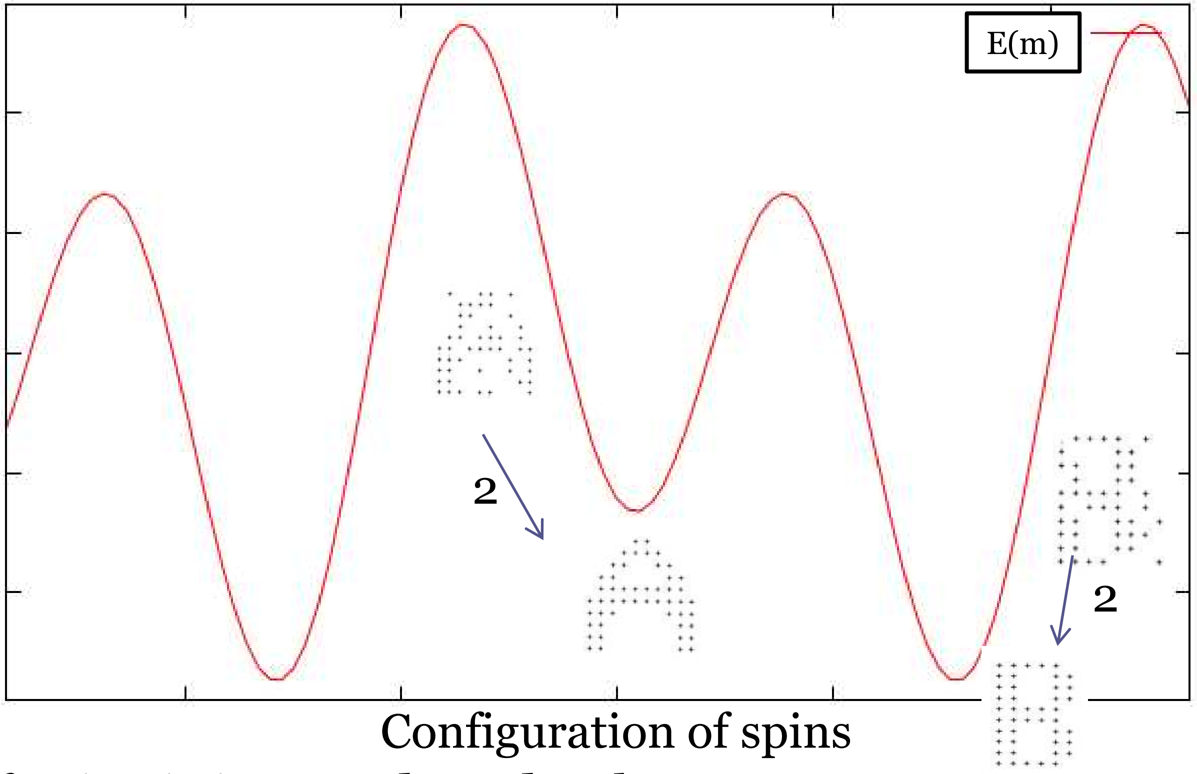
- To store a particular state m of the neural network in memory, the energy of that configuration of spins must be set as a local minimum by:

$$J_{i,j} = S_i(m) S_j(m)$$ to store one configuration

$$J_{i,j} = \frac{1}{M} \sum_m S_i(m) S_j(m)$$ to store multiple configurations

Numerical Approach: Monte Carlo Method

$$E_{network} = -\sum_{i,j} J_{i,j} S_i S_j$$



1. Initial configuration of spins is inputted randomly.
2. Using a time step function, every spin in the lattice is given a chance to flip:

   Resulting ΔE is negative $\Longrightarrow$ Flip the spin

   Resulting ΔE is positive $\Longrightarrow$ Leave the spin unchanged
3. The lattice eventually reaches a stable spin configuration, an energy local minima.

Testing and Numerical Experiments

- Determine the degree of randomness that can be present in the initial configuration of spins for it to eventually reach one of the stored configurations. This will be tested chronologically with the number of stored configurations increasing incrementally.
- Determine the minimum Hamming distance (difference between 2 configurations) required for our simulation to yield consistent results.

Project Timeline

October 22nd	Project Proposal and Presentation
October 22nd – 25th	Complete Research and Code Design
October 28th – November 1st	Code Implementation/Testing
November 4th – November 8th	Code Implementation/Testing
November 11th – November 15th	Conduct Numerical Experiments
November 18th – November 22nd	Analyze Data, Begin Final Report
November 25th – December 2nd	Complete and Submit Final Report

References

- "Basic Concepts for Neural Networks. "Cheshire Engineering Corporation. N.p., n.d. Web. 21 Oct. 2013. <http://www.cheshireeng.com/Neuralyst/nnbg.htm>.

- "Basic Concepts for Neural Networks. "Cheshire Engineering Corporation. N.p., n.d. Web. 21 Oct. 2013. <http://www.cheshireeng.com/Neuralyst/nnbg.htm>.

- Nelson, Philip Charles, Marko Radosavljević, and Sarina Bromberg. Biological physics: energy, information, life. New York: W.H. Freeman and Co., 2004. Print.

Diffusion-limited Aggregation

PHYS 210 Term Project Proposal

Jingjing Le

Oct 2013

Overview

- Diffusion-limited aggregation is the process whereby particles undergoing a random walk. The particles cluster together to form aggregates of such particles.
- It can model systems such as snowflake growth, lichen growth, lightening path, mineral deposition, electrodeposition and so on.

Project Goals

- To write a Matlab code which simulates and visualize the diffusion-limited aggregation.
- To investigate the effects of parameters on the model

Mathematical Formulation

- Fractal growth

If N(r) denotes the number of particles within radius r of a circle (or sphere) encircling a DLA then **$N(r)= kr^{d}$**

- Density-Density correlation

$$C(r)=N^{-1}\sum_{r'}\rho(r)\rho(r+r')$$

C(r): average density   r: distance separating two sites.

- Correlation:

$C(r)\ \alpha\ r^{a}$

$d = 2 - a$ (in 2D simulation)

Mathematical Formulation

- Gauge field Ψ on the surface, which grows by deposition of diffusing particles

$$\frac{\partial \Psi}{\partial t} + (\vec{V} \cdot \vec{\nabla})\Psi = 0$$

$\vec{V}$ ($\vec{r}$, t) is the velocity field

- Given rate of absorption per unit time Φ($\vec{r}$, t)

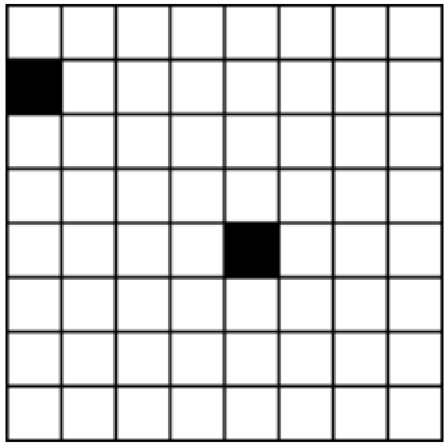
$$\frac{\partial n}{\partial t} = D\nabla^2 n - q - \lim_{\epsilon \to 0^+} \Phi(\vec{r}, t)\delta(\Psi - \epsilon)|\vec{\nabla}\Psi|$$

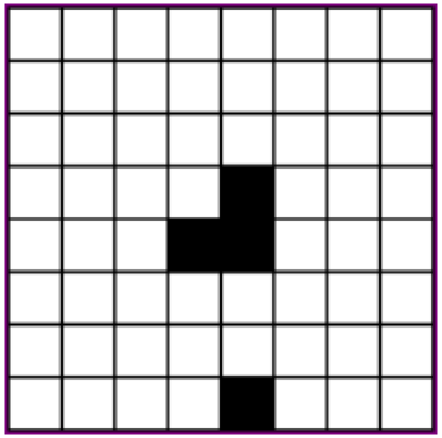
- description of the surface growth

$$\frac{\partial \Psi}{\partial t} - \lim_{\epsilon \to 0+} D \int dt' d\vec{r}' \vec{\nabla} G(\vec{r} - \vec{r}', t - t') \cdot \vec{\nabla}\Psi(\vec{r}, t) \frac{\partial \Psi}{\partial t'}(\vec{r}', t')\delta[\Psi(\vec{r}', t') - \epsilon] = -\frac{1}{n_0} D\vec{\nabla} n^s \cdot \vec{\nabla}\Psi$$

Simulation Approach

- using a lattice
- An initial seed particle at the centre
- Second particle somewhere on the lattice, moving in random motion (in 2D square lattice: up, down, left, right)
- stops when it meets the seed particle or move out from the lattice
- Repeat the steps by introducing another





Testing

- Restrict the number of particles to under 10 to test if the simulation works as expected

Exploring Variations

- Introducing sticking coefficient

  -- when a particle reaches the cluster, the probability of attaching

  -- from 0 to 1 (0: never attaches, 1: always attach)
- Varying initial positions of the seed particle

  -- this might alter the shape from the simulation
- Changing the geometry of the starting seed

  -- e.g. using line instead of point as the starting seed

Project Timeline

Dates	Activities
10.21-10.27	Basic research, begin code design
10.28-11.14	Implement code
11.15-11.18	Test code
11.19-11.26	Experiments, analyze, begin report
11.26-11.30	Finish report
12.1	Submit project

Reference

- Diffusion-limited aggregation, T.A. Witten and L.M. Sander, Phys. Rev. B 27, 5686-5697
- http://classes.yale.edu/fractals/panorama/physics/dla/dla.html

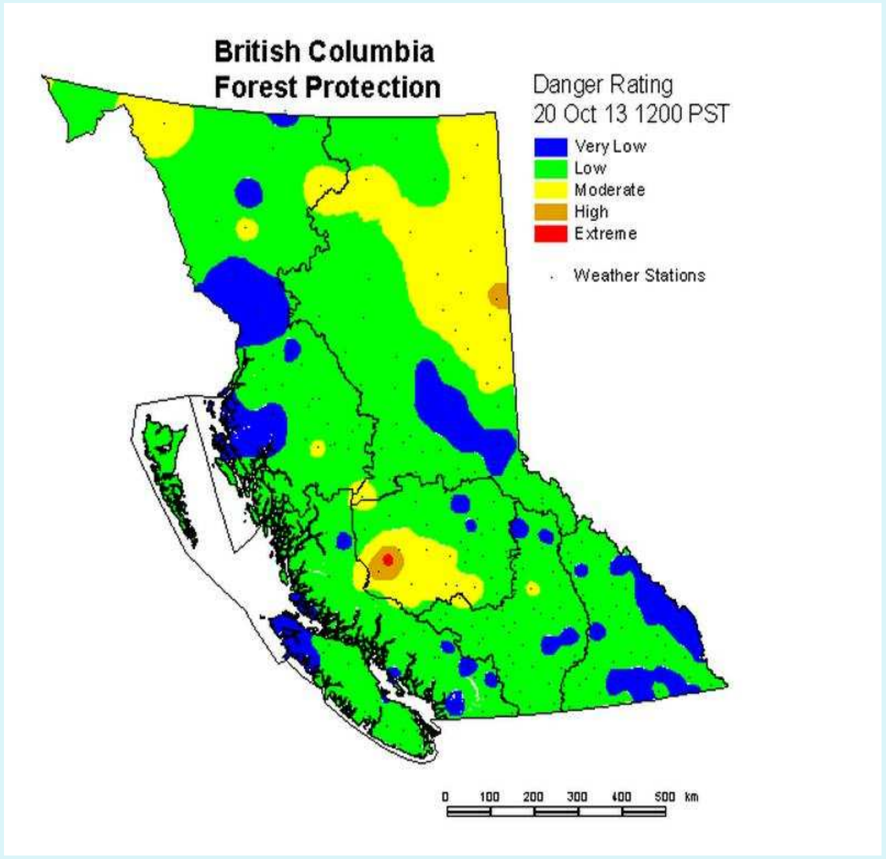
PHYS 210 Project Proposal

FOREST FIRE MODELING USING CELLULAR AUTOMATA

Prepared by Ariel Lee

Overview of Project

- Forest Fire starts and spreads **rapidly**
- Faster spread if there is wind
  - STOCHASTIC PROCESS = process in nature where we can analyze using probability
- Forest fires spread depending on if its adjacent trees are burning
- **CELLULAR AUTOMATA** = collection of cells on a specific grid that evolves over time with respect to rules based on the situation of nearby cells



Assumptions

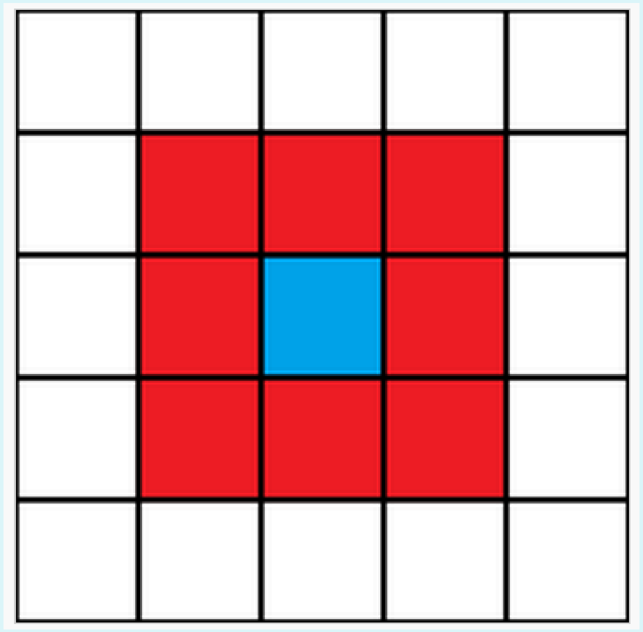
- For simplicity, ignore all wind factors (although in reality, wind causes forest fires to spread more quickly and randomly)
- As well, assume the terrain is completely flat, thus allowing for simple 2-D Modeling
- Lastly, assume no significant obstacles (no external forces other than lightning to ignite fires)

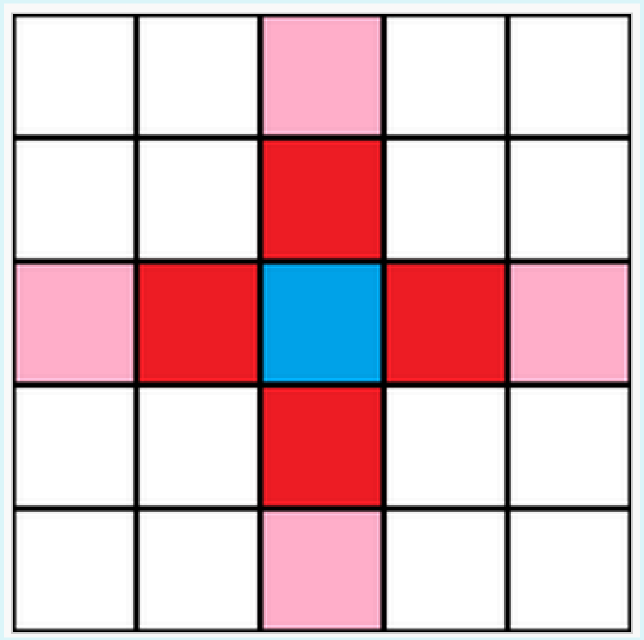
Project Goals

- To simulate forest fires using **finite cellular automata** and 2-D modeling within Matlab (Octave)
- To observe and examine the spreading of forest fires
- To predict where the forest fire may spread to and if so, at what rate

Visualization: Cellular Automata

- Cellular Automata can be displayed in two ways:


Moore Neighborhood


Von Neumann Neighborhood

Important Rules

- All cells are in one of the four following states:
  1) Ignitable Cell (normal tree, can be ignited by lightning/other factors)
  2) Burning Cell (caught on fire)
  3) Burnt Cell (no more fire, completely dead)
  4) Growing Cell (can evolve into a new tree)

- This cycle continues on perpetually
- Again, trees are only “burnable” if its neighbors are also ignited (depends on adjacent trees)

Mathematical Formulae

- State transition function rule :

$$f \in F$$

$$S_c^{t+1} = f(S_c^t, S_{n(c)}^t)$$

S_c^{t+1} = state of new cell

S_c^t = state of current cell

$S^t_{n(c)}$ = state of neighborhood cell

Also: $\frac{\text{A(area burnt)}}{\text{A(total Area)}}$ ~Intensity of Fire

Numerical Approach and Testing

- Using hexagonal cellular automata, find number of time steps it takes for forest fire to spread
- Examine the time it takes for a forest fire to spread within one region
- Initialize forest fire at various locations to observe the effects of being in a central region or along the coast
- Increase the origin of forest fires at different regions and due to various causes (lightning strike/abundant forest fuel)
- Run tests on how multiple forest fires affect each other and change the rates at how quickly it spreads

PLOTTING TOOLS = Matlab, 2D Modeling and Hexagonal Cellular Automaton

Schedule for Project

DATES	**Assignments**
Oct 20 ~ Oct 29	Research, Read up! Begin learning code for project
Oct 30 ~ Nov 15	Implement Code
Nov 16~ Nov 19	Test Code
Nov 20 ~ Nov 25	Run Experiment, Start writing report
Nov 26 ~ Nov 29	Complete Report
Nov 29	HAND IN REPORT
Dec 2	**ABSOLUTE DEADLINE**

Websites Visited

THANK YOU!

http://bcwildfire.ca/weather/maps/danger_rating.htm

http://bcwildfire.ca/

http://www.jstor.org/discover/10.2307/2302572?uid=3739400&uid=2&uid=3737720&uid=4&sid=21102804457663

http://mathworld.wolfram.com/CellularAutomaton.html

http://bib.irb.hr/datoteka/278897.Ljiljana_Bodrozic_ceepus2006_2.pdf

Toomre Model of Galaxy Collisions

Physics 210 Project Proposal

Oct 22, 2013

Chris Mann

Overview

- Like all massive bodies, galaxies are drawn together by gravitational attraction.
- When galaxies inevitably meet, they do not “collide” in the sense of two objects hitting one another. Their very low density causes them to merge and redistribute their relatively light constituents.
- This model will be simplified by ignoring dark matter, and considering all the mass to be concentrated at the galactic core.

Project Goals

- To write Matlab code which simulates and visualizes the merger of two galactic bodies.
- To determine how initial conditions affect the final outcome of the merger:
    - Two spirals -> ???
    - Two ellipticals -> ???
    - Spiral + elliptical -> ???
    - How does angle of approach affect outcome?

Mathematical Formulation

- The following formulae will be important:

$$F = G\frac{m_1 m_2}{r^2}$$

$$F = ma_c = \frac{mv^2}{r}$$

$$P^2 = \frac{4\pi^2}{G(M_1 + M_2)}a^3$$

$$f'(t_j) \simeq \frac{f_{j+1} - f_{j-1}}{2\Delta t}$$

$$\left(\frac{P}{2\pi}\right)^2 = \frac{a^3}{G(M+m)}$$

$$f''(t_j) \simeq \frac{f_{j+1} - 2f_j + f_{j-1}}{\Delta t^2}$$

ASSUMPTIONS

- The model is greatly simplified by assuming:
  - The constituent stars do not produce a gravitational force (however, they may need to be assigned a mass in order to interact with the system)
  - All of the gravitational force in the system is generated by the mass of the galactic cores.
  - There are no physical collisions (ie. the galaxies pass through one another and will only be affected gravitationally)
  - The merger occurs in 2-dimensions

Numerical Approach

- Will consider the 2-dimensional vectors for position, velocity, and acceleration.
- This information will be iterated through equations of force and motion to determine where each particle will be in the following timestep.
- As the size of the timesteps approach zero, the simulation should become more and more accurate.

Visualization

- The 2-dimensional position coordinates of galaxy cores and constituent stars will be plotted for each specific time t_j.
- Playing many of these plots from t_o to t_{max} will create an animated simulation of the events of the merger over time.

Testing & Numerical Experiments

- Vary conditions for each galactic galaxy:
  - Mass of core, velocity, angle of approach, size
- Gradually increase number of timesteps to as large as feasible to improve accuracy.
- Determine what initial conditions produce single galaxies, satellite galaxies, or galaxies altered in various ways

Project Timeline

Date	Task
10/15 – 10/26	Do basic research, derive equations & begin code design
10/27 – 11/15	Implement code
11/16 – 11/19	Test code
11/20 – 11/26	Run numerical experiments, analyze data, begin report
11/27 – 11/29	Finish report
11/29	Submit project (absolute deadline 12/02)

References

- Images:
  - Google Images
- Wikipedia:
  - http://en.wikipedia.org/wiki/Galaxy_merger

Equilibrium Configuration of N Identical Charges on the Surface of a Sphere

Physics 210 Term Project Proposal

Dylan Marquette

Overview

- Equal charges will repel each other with a force that can be found using Coulomb’s law.
- When a number of equal charges are placed on a sphere the forces that they exert on each other will move them into an equilibrium position
- This equilibrium position is the configuration with the lowest energy
- There will be a new equilibrium configuration for ever additional charge

Project Goals

- To design a MATLAB program that models the distribution of charges on a sphere

- Investigate a variety of solutions to initial conditions, such as the number of charges and their initial positions

Mathematical Formulation

- For a system of discrete charges this formula may be used to find the force acting on a particle:

$$\boldsymbol{F}(\boldsymbol{r}) = \frac{q}{4\pi\varepsilon_0}\sum_{i=1}^{N} q_i \frac{\boldsymbol{r}-\boldsymbol{r}_i}{|\boldsymbol{r}-\boldsymbol{r}_i|^3} = \frac{q}{4\pi\varepsilon_0}\sum_{i=1}^{N} q_i \frac{\widehat{\boldsymbol{R}_i}}{|\boldsymbol{R}_i|^2},$$

- Coulomb's law stated as a mathematical expression:

$$\boldsymbol{F}_{\mathbf{12}} = k_e \frac{q_1 q_2 \hat{\mathbf{r}}_{\mathbf{21}}}{r^2}$$

Additional Information

- All the charges will be identical
- Charges are restricted to the surface of a sphere

Project Timeline

Dates	Activities
Oct. 22 – 27	Research , derive equations , begin code design
Oct. 28 - Nov. 9	Implement code
Nov. 10 - 12	Test code
Nov. 13 - 20	Run numerical experiments, start report
Nov. 21 – Dec. 2	Finish and submit report

References

- http://en.wikipedia.org/wiki/Thomson_problem
- http://en.wikipedia.org/wiki/Coulomb%27s_law
- http://tracer.lcc.uma.es/problems/thomson/thomson.html

Equilibrium Configuration Identical Charges on the Surface of a Sphere

Phys 210 Term Project Proposal

By Daniel Mclean

October 22 2013

Overview

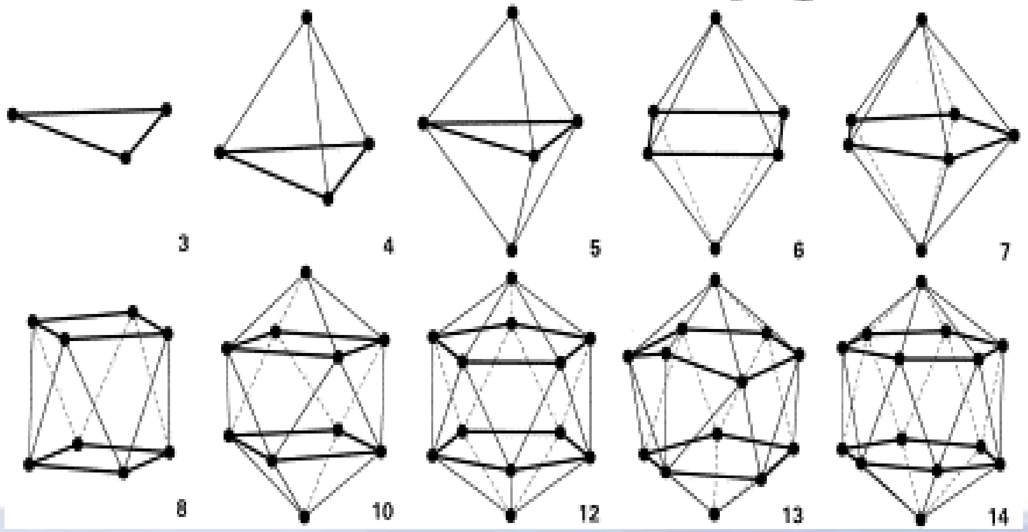
- To find the minimum potential energy to arrange of a given number of charged particles on the surface of a sphere by maximizing their distance
- All charges will be identical and repel each other
- If another charge is added will find a new equilibrium alignment

Goals

- Make a simulation with MATLAB to see how N charges distribute themselves on the surface of a sphere
- Use Finite difference approximations to solve differential equations
- See how the some N have more than one configuration

Formulas

- Coulomb's Force
  - $$|\boldsymbol{F}| = k_e \frac{|q_1 q_2|}{r^2}$$
- Electric Potential Energy
  - $$U_E(r) = k_e q \sum_{i=1}^{n} \frac{Q_i}{r_i}$$



Project Timeline

Dates	Activities
Oct 15 – Oct 26	Do basic research, derive equations and begin code design
Oct 27 – Nov 15	Implement code
Nov 16 – Nov 19	Test code
Nov 20 – Nov 26	Run numerical Experiments, analyze data,begin report
Nov 27 – Nov 29	Finish report
Nov 29	Submit project!

References

- http://www.mathpages.com/home/kmath005/kmath005.htm
- http://en.wikipedia.org/wiki/Coulomb's_law
- http://en.wikipedia.org/wiki/Electric_potential_energy
- http://tracer.lcc.uma.es/problems/thomson/thomson.html
- https://www.google.ca/
- http://laplace.physics.ubc.ca/210/Doc/term-projects/kdv.pdf

N-Body Problem: Recreating a Rogue star formation

Haydn Melin

October 2013

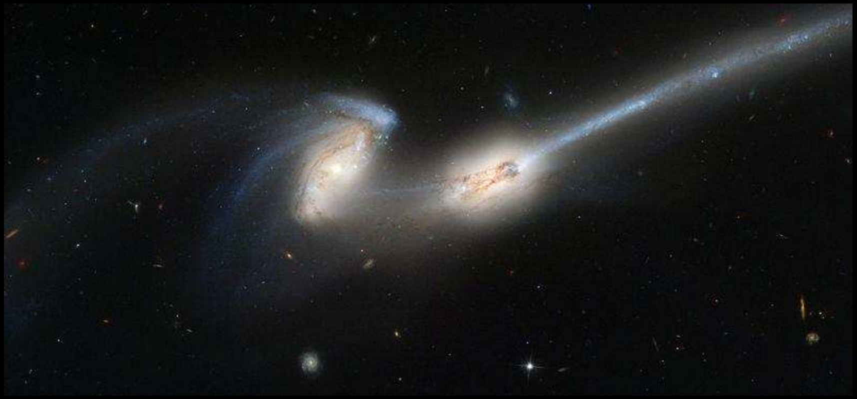
Overview
Basic questions

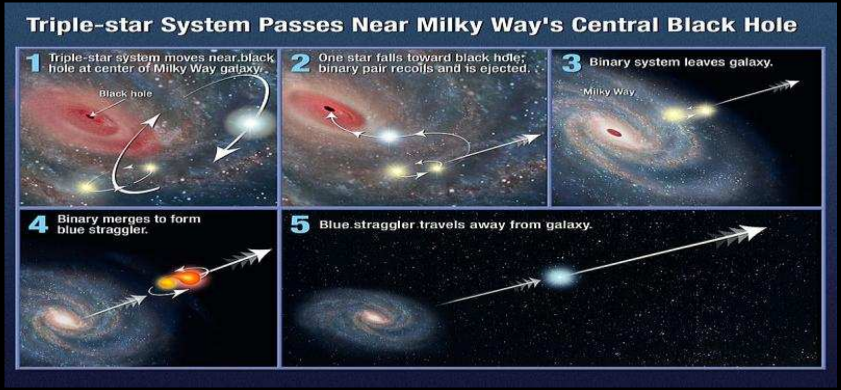
- What is a rogue a.k.a. intergalactic Star?

- How do they form?

- How does this relate to the N-body problem?

Cont.




Triple-star System Passes Near Milky Way's Central Black Hole
1 Triple-star system moves near black hole at center of Milky Way galaxy.
Black hole
2 One star falls toward black hole; binary pair recoils and is ejected.
3 Binary system leaves galaxy.
Milky Way
4 Binary merges to form blue straggler.
5 Blue straggler travels away from galaxy.

- The common belief was that stars only existed in galaxies. This was disproved in 1997 with the discovery of intergalactic stars in the Virgo cluster of galaxies.
- The top picture as a theory how rogue stars form (galazy collisions)
- The bottom picture is another theory.
- My simulation will be closer to the theory below but, will have a binary star system, not a triple-star system.

Goals

- To design a MATLAB (Octave) code that will simulate the N-body problem
- To see that if initial conditions are changed, such as mass and velocity, energy and momentum are conserved within an uncertainty.
- To see the outcome of the simulation when certain properties are changed
- To successfully simulate my interpretation of a rogue star formation

Formulas

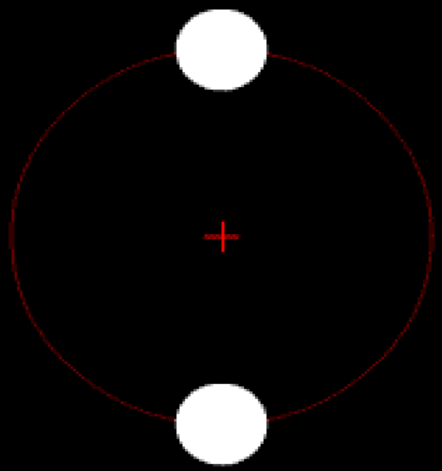
- Newton's second law
- $F_i = M_{i*}d^2x_i/dt^2 = M_{i*}dv_i/dt = m*a$
- Where i=1,2,3…n bodies
- Newton’s gravitational law
- $F_{12} = -G*m_{1*}m_2/r^2$
- Where $r = r_1(1+r_2/r_1)$ or $r = r_1(1+m_1/m_2)$
- $F = Gm_{2r}m_1/(r_1{}^2)$
- Where $m_{2r} = m_1{}^2*m_2/(m_1+m_2)^2$

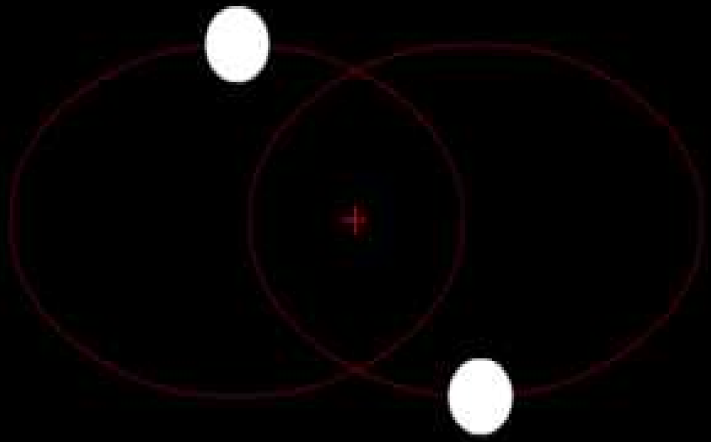
Approach to solving

- Using the FDA’s which were given in class I would like to determine the force and acceleration of each mass (star) before and the velocity of the rogue star after

Numerical experiments

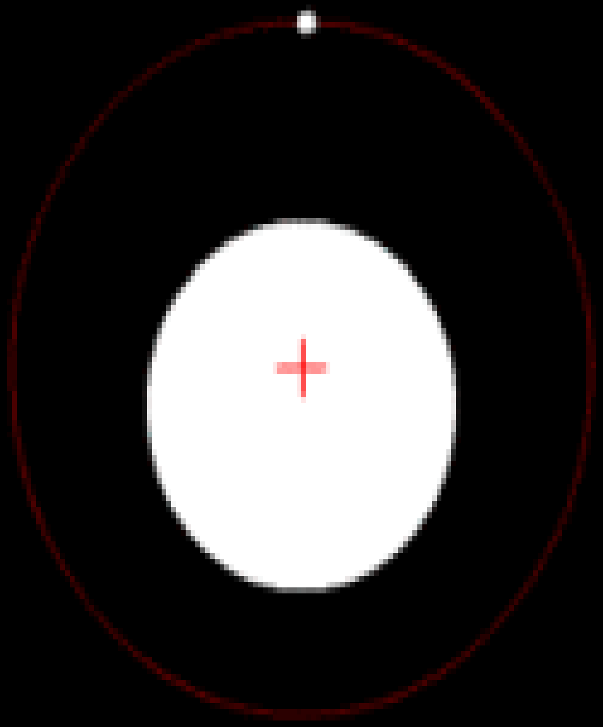
- Will start simulation with mass being the same and with equal distance away from the CoM
- Will change masses therefore changing the CoM position and the orbits of the 2 masses





Cont.

- My end goal is to get something similar to this
- Where the big mass is the singularity (black hole) and the smaller mass is the 2 smaller masses
- Note, I should try to make the black hole's mass so large that it essentially does not rotate around the CoM and when one of the stars falls into in, the black hole does not change mass



Timeline

Date	Activities
Oct.24-Oct.31	Do basic research, derive equations
Nov.1-Nov.9th	Design and write the code
Nov.11-Nov.15	Implement code
Nov.16-Nov.23	Test code and debug code
Nov.18-Nov.22	Begin numerical experiments and analyze data
Nov.24	Start report
Nov.28	Finish report
Nov.29th	Double check and submit report

Fun simulation

- http://janus.astro.umd.edu/orbits/nbdy/rstar.html
- Questions?

References

- http://en.wikipedia.org/wiki/Binary_star
- http://astro.unl.edu/naap/ebs/centerofmass.html
- http://en.wikipedia.org/wiki/Intergalactic_star
- http://janus.astro.umd.edu/

Forest fire modeling using cellular automata

Samuel Andres Navarro Leiva

Overview

- Forest fire simulations help plan wildfire operations. A widely used forest fire simulator is FARSITE: Fire Area Simulator.
- Forest fire simulations can be used to predict the impact of this natural disturbance with respect to its frequency.

Project Goals

- Write a MATLAB code that simulates a forest fire with respect to time using a cellular automaton.
- Compare my simulation with the many others available from internet sources in order to judge the correctness of my own code.
- Investigate the effects that other variables would have on a forest fire (e.g., humidity).

Visualization:

- 200x200 cellular automaton.

Mathematical Formulation:

- Each cell (or grid point) is surrounded by other cells.
- In each unit time the probability of the fire crossing to an adjacent cell is given by p.
- p is usually equal to 0.5

Project Timeline

Dates	Activities
10/20 – 10/28	Research necessary mathematics & begin code design
10/28 – 11/5	Implement code
11/5 - 11/13	Test code
11/13 – 11/21	Run experiment, analyze data, begin report
11/21 – 11/29	Finish report
11/29 – 12/01	Revise code & report, submit report

Simulation of gravitational forces on N-bodies in a plane

PHYS 210 Term Project Proposal

James Nightingale

-Overview

-A given number of point masses will exert a gravitational attraction on each other, by ignoring other forces, it is possible to model the motion that would result.

-This is similar to the behaviour of systems such as stars and planets in outer space, which have large distances, and an absence of other significant forces.

-Project Goals

-To use Matlab to create a program capable of solving the differential equations of motion for various initial positions and numbers of bodies Possibly including both random patterns and those of real astronomical entities (eg. binary stars)

-To display the results of the calculations for different situations graphically and test the correctness of the program's implementation through approximations.

Mathematical Formulation

- $F=m_i(dr_i2/dt^2)$
  For the equation of motion in conjunction with the equation for the gravitational force, to be solved by means of finite difference approximation.

Project Timeline

-The time line is still indeterminate.

Ray Tracing in 2D Space

Simon Norman-Hobbs

Physics 210
Fall 2013

Overview

- Classically a light source can be approximated as a group of rays instead of a wave
- The visualization of an optical system is greatly simplified through the use of rays
- Ray tracing is a graphical method of studying the light interactions as rays with components in an optical system.

Purpose

- To study an optical system with input objects of a specific user defined position, orientation and specifications
  - Source
    - Wavelength, convergence/divergence/coherent
  - Mirrors
    - Convergent/divergent, Focal length
  - Prisms
    - Length, apex angle, Refractive index
  - Irises
    - Length, gap
  - Lenses
    - Refractive index, Focal Length

Mathematical Model

- Thin Lenses
  - Using a thin lens approximation the lens equation is simplified to the lens makers equation for f >> lens thickness:

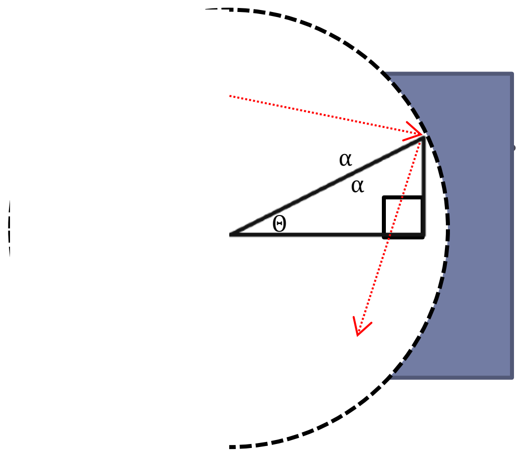
$$\frac{1}{f} \cong (n-1)\left[\frac{1}{R_1} - \frac{1}{R_2}\right]$$

    - Using a ray transfer matrix gives the reflected new angle α_2 and height from optical axis h:

$$\begin{pmatrix} h \\ \alpha_2 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ \frac{1}{f} & 1 \end{pmatrix} \begin{pmatrix} h \\ \alpha_1 \end{pmatrix}$$

- Spherical Mirrors
  - Angle of incidence equal angle of
    - For a given radius R, and a height from the optical axis 'h', the reflected angle α can be found using a ray transfer matrix such that:

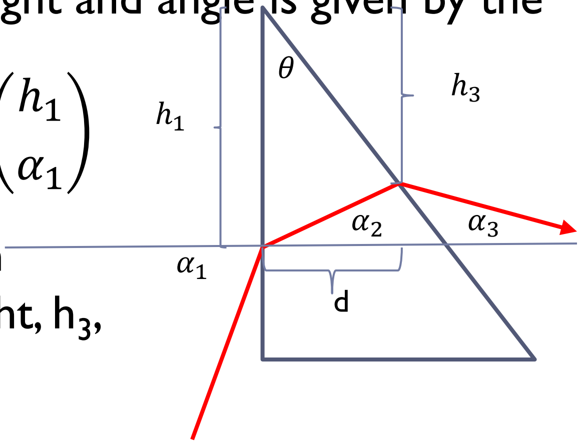
$$\begin{pmatrix} h \\ \alpha_2 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ \frac{R}{2} & 1 \end{pmatrix} \begin{pmatrix} h \\ \alpha_1 \end{pmatrix}$$



Mathematical Model

- Right Prisms
  - Given a specific index of refraction for a given wavelength η, the reflected refracted beam is given by the combination of two ray transfer matrixes.
    - The first interaction to find the transformed height and angle is given by the transfer matrix:

$$\begin{pmatrix} h_2 \\ \alpha_2 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & \frac{\eta_1}{\eta_2} \end{pmatrix} \begin{pmatrix} h_1 \\ \alpha_1 \end{pmatrix}$$



    - To account for the change in height of the beam in its propagation through the prism, the new height, h_3, Is given by $h_3 = h_2 + d \cdot tan(\alpha_2)$

    - The second transfer matrix gives the new angle of the ray as it exits the prism

$$\begin{pmatrix} h_3 \\ \alpha_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & \frac{\eta_2}{\eta_1} \end{pmatrix} \begin{pmatrix} h_2 \\ \alpha_2 \end{pmatrix}$$

Timeline

Dates	Task
10/19-10/25	Topic Research, Code Layout
10/26-11/09	Writing Code
11/10-11/14	Hand Test Code Calculations and Tracing
11/15-11/22	Write Report
11/23-12/02	Make Minor Changes

References

- Hecht, Eugene (1987). *Optics*, Addison Wesley
- http://en.wikipedia.org/wiki/Ray_transfer_matrix_analysis

Questions: Ray Transfer Matrixes

- The definition of a ray transfer (RT) matrix is:

$$\begin{pmatrix} h_2 \\ \alpha_2 \end{pmatrix} = \begin{pmatrix} \frac{h_2}{h_1} & \frac{h_2}{\alpha_1} \\ \\ \frac{\alpha_2}{h_1} & \frac{\alpha_2}{\alpha_1} \end{pmatrix} \begin{pmatrix} h_1 \\ \alpha_1 \end{pmatrix}$$

- The RT matrixes for each optical object is just derived from how each component changes with respect to each other.

Modeling Wildfire Spread and Containment

Katie Olfert

Physics 210

Goals

- To create a matlab simulation of a wildfire
- To incorporate such things as the way it spreads, and the way it is affected by water
- To gain valuable knowledge about programming
- To not fail this course

WILL DO

- Wind
  - (direction/speed)
- Water
  - (waterbombers)

Will Ignore/Assume

- Some Weather Aspects
  - Temperature
  - Moisture
- Change in Altitude = 0
- “Fuel Load” Constant

The Concept:

- The map will be split into cells.
- Each cell has one of the following states:
  - Unburnt: Has not had contact with fire
  - Burning: Cell is on fire
  - Burnt: Burn-time delay has passed, this cell is “dead”
  - Burning-wet: If water is dropped on this cell, will transition to “burnt-wet” or “unburnt-wet”
  - Unburnt-wet: If burn-time delay exceeds water-time delay. This cell cannot burn again
  - Burnt-wet: If burn-time finishes before the water delay. This cell is now “dead”

Some Relevant Equations

Rothermel BEHAVE model

$$R = \frac{I_R \xi (1 + \phi_w + \phi_s)}{\rho_b \varepsilon Q_{ig}}$$

I_R - reaction intensity

ξ - no wind propagation flux ratio

ϕ_w - additional propagating flux produced by wind

ϕ_s - additional propagating flux produced by slope

ρ_b - bulk density (fuel particle density)

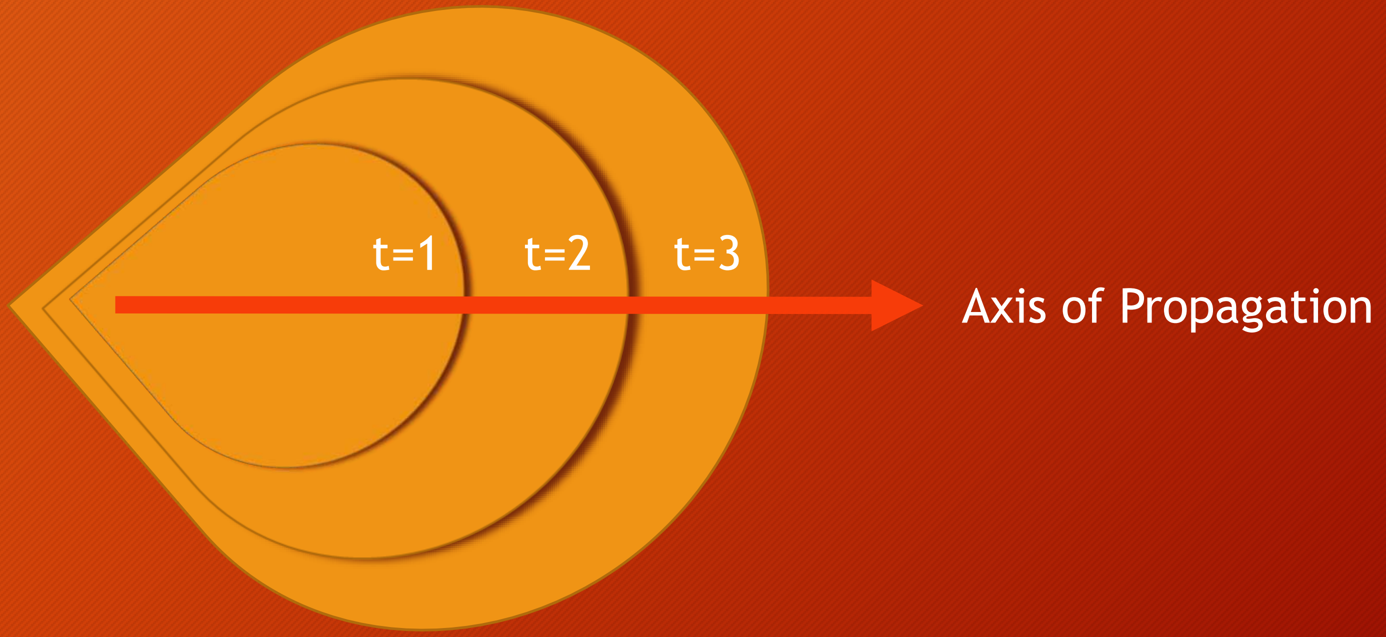
ε - effective heating number

Q_{ig} - the heat of preignition (the energy per unit mass required for ignition)

http://www.ecs.umass.edu/mie/faculty/smith/StepanovJMSmith_EJOR_R4.pdf

Some Relevant Equations

- Fire Shape:
    - Length/Width = 1 + .25 * (windspeed)

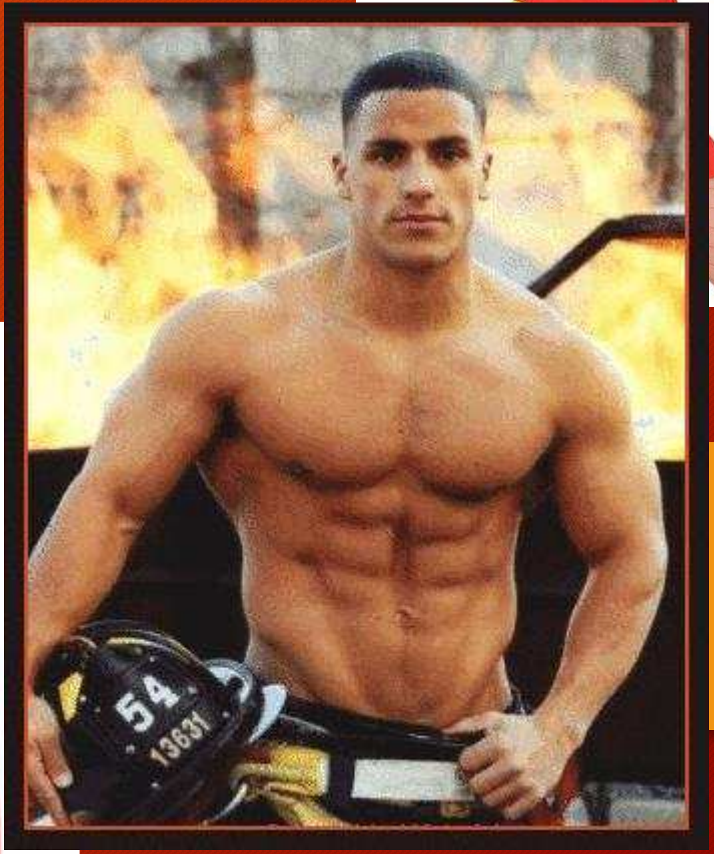


Honest Project Timeline

Task	Date
Presentation	October 24th
Get Started on Improving design	October 25th
Start coding	December 1st
Finish Coding and debug	December 2nd, 11:59 PM

THANK YOU FOR LISTENING

Questions?



Time Dependent Schrödinger Equation in 1D

Ben Pearce

The weight factor may
function.
a certain wa
$i\hbar\, d\psi/dt = H\psi,$
itian (or self-adjoint) linear op
n expressed in the representat
eralization of the Hamilton-J

PHYS 210 Term Project Proposal

Overview

- The Schrödinger equation is a linear partial differential equation describing how the quantum state (i.e. the wave function, i.e. the analog to position in classical mechanics) of a physical system changes over time.
- It takes the form: $i\hbar * \partial/\partial t(\Psi) = \hat{H}\Psi$,

  where Ψ is the wave function and $\hat{H}$ is the Hamiltonian operator, which describes the total energy of a given wave function (i.e. to make this equation less daunting, they invented a letter to say: this corresponds to a more complex equation that depends on your scenario).

Project Goals

- Write MATLAB code to solve the time-dependent Schrödinger equation in 1D using a finite difference approximation and use that algorithm to simulate how a quantum particle in a box propagates over time **and** how a “much” of a quantum particle makes it through various barrier thicknesses (i.e. quantum tunneling).
- Test for correctness via convergence testing.

Mathematical Formulation

- For 1 particle in 1D, that is dependent on time, the Hamiltonian takes the form:

  $\hat{H} = -(\hbar^2/2m)*\partial^2/\partial x^2 + V(x)$

  where V(x) is the potential energy

- For simplification, the Schrödinger equation is just a form of the heat equation: $u_t = \alpha^2 u_{xx}$

  With boundary conditions $u(0,t) = 0 = u(L, t)$,

  initial condition $u(x, 0) = u_0(x)$

  on the domain $0 \leq t \leq t_{max}$ and $0 \leq x \leq L$

Numerical Approach

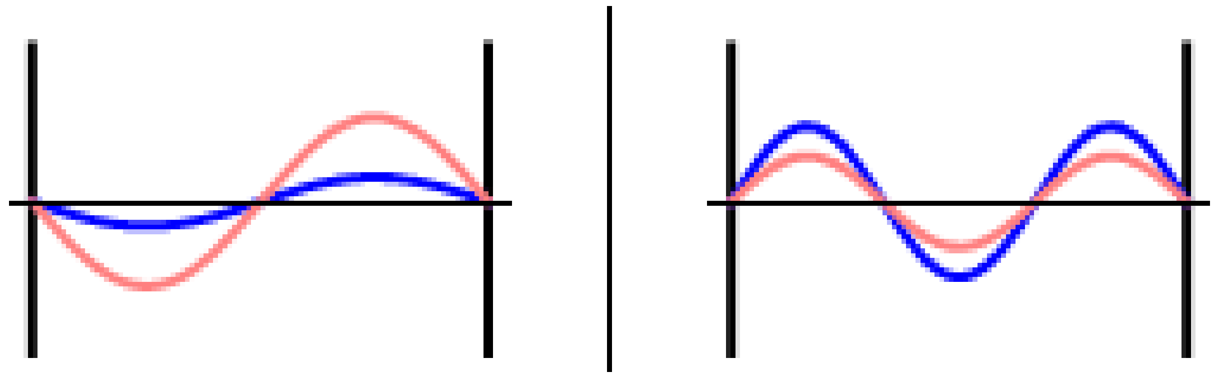
- I will be using the Crank-Nicolson Finite Difference Approximation, because it is quadratic in error over space and time ($O(\Delta x^2)$ and $O(\Delta t^2)$) whereas other methods (implicit, explicit) are linear in error over time ($O(\Delta t)$).

$$\frac{u_j^{n+1} - u_j^n}{k} = \frac{1}{2}\left(\frac{u_{j+1}^{n+1} - 2u_j^{n+1} + u_{j-1}^{n+1}}{h^2} + \frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{h^2}\right)$$

n = position on time axis
j = position on space axis
k = Δt
h = Δx

Visualization and Plotting Tools

- MATLAB plotting facilities with FOR loops to simulate the propagation of the wave function.
- There may be a better way to animate this; more research is necessary.



Testing and Numerical Experiments

- Convergence testing over increasing discretization to test quadratic convergence behavior: Δx = (xmax- xmin)/2l, for l = 0, 1, 2, …, 13, 14, 15, …
- Numerical and visual testing with various particle energy eigenstates.
- Numerical and visual testing with various barrier thicknesses.

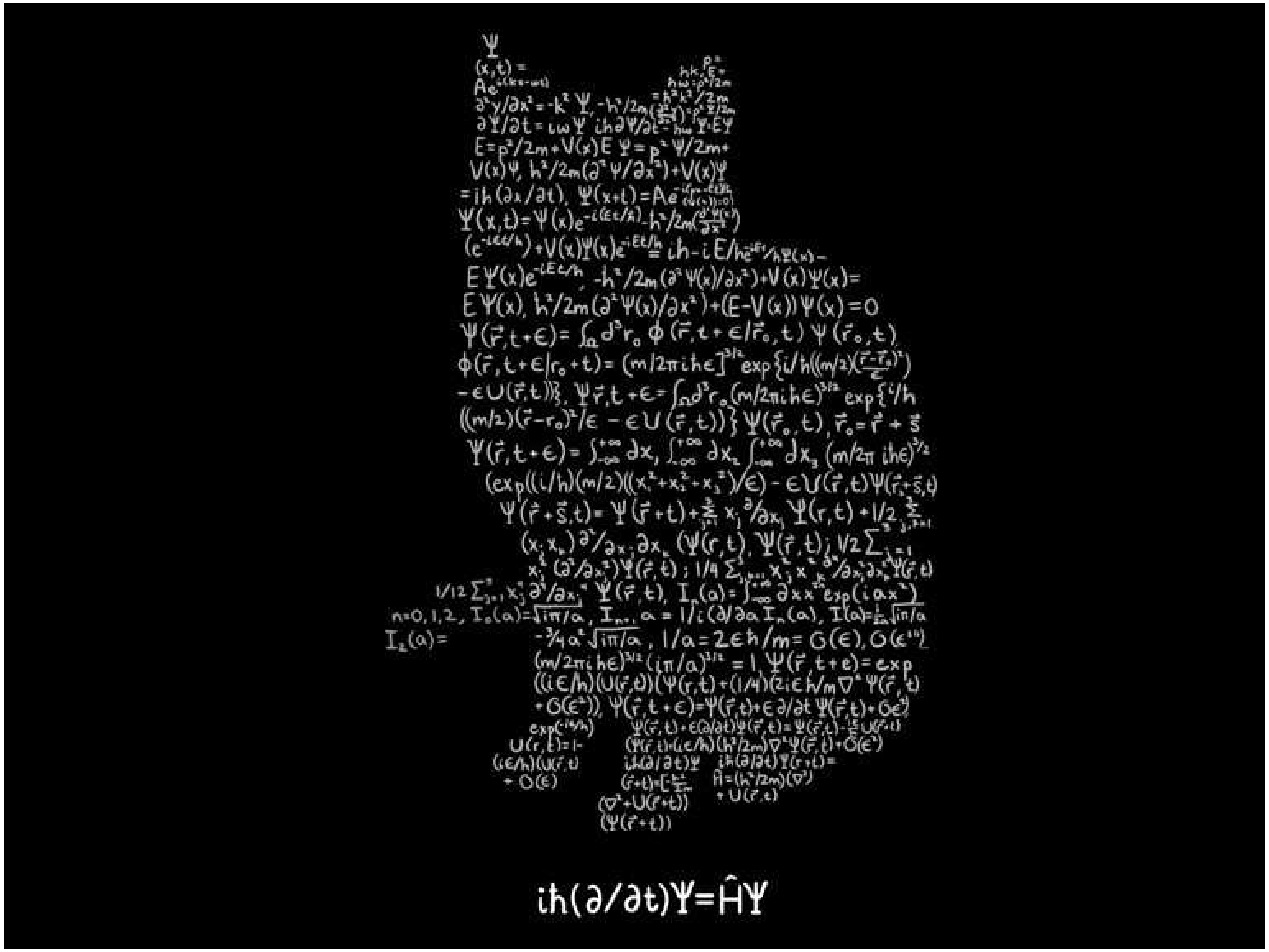
Project Timeline

Date	Task
October 7th – 28th	Plan Project
October 29th – November 15th	Write Code
November 16th – November 19th	Test Code
November 20th – November 24th	Run numerical experiments and analyze simulations
November 25th – November 30th	Write Report
December 1st	Submit Report

References

- http://en.wikipedia.org/wiki/Finite_difference_method
- http://www.amphioxus.org/sites/default/files/MATLAB-animatedPlotsTutorialArmin.pdf
- http://en.wikipedia.org/wiki/Particle_in_a_box
- http://en.wikipedia.org/wiki/Schr%C3%B6dinger_equation
- http://hyperphysics.phy-astr.gsu.edu/hbase/quantum/barr.html
- http://www.123rf.com/photo_8085180_the-schrodinger-equation.html
- http://thinng.com/4447-haunted-scarecrow

Questions/Comments?



Toomre Model for Galaxy Collisions

Physics 210 Term Project Proposal

2013 Fall

Chelsea Powrie

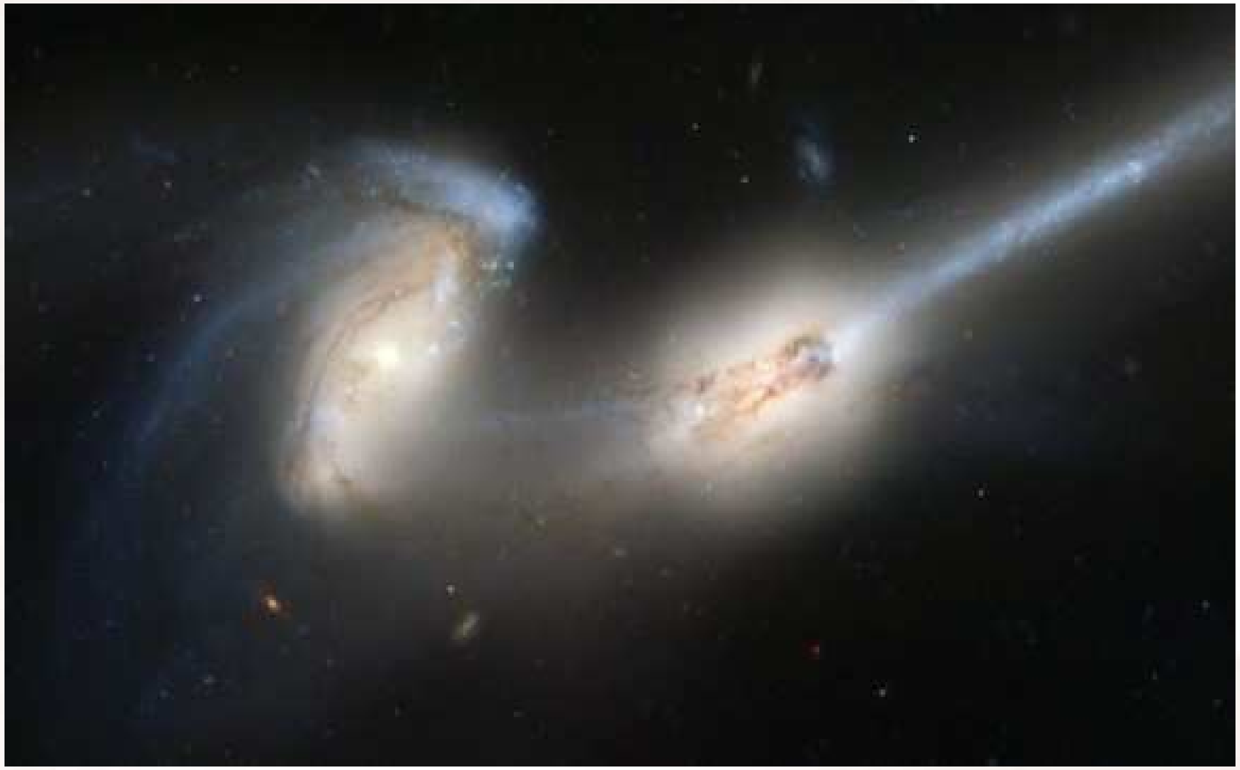
Overview

- The Toomre Model simplifies the calculation of galactic gravitational interaction
- It assumes masses of individual stars to be negligible in comparison to the overall mass of the galaxies
- Stars in these galaxies are approximated by particles

Project Goals

- Write a functioning MATLAB code which is capable of performing the mathematics involved and predicting outcomes given varied initial parameters
- Use Newton's version of Kepler's third law equation for the calculation of orbital periods
- Create a short video simulation of a collision

- Hopefully be able to recognize in my results characteristics noticeable in real life galaxy collisions, such as NGC 4676 “The Mice”



Mathematical Formulations

$$P^2 = \left[\frac{4\pi^2}{G(m_1 + m_2)} \right] a^3$$

$$P = sidereal\ \ period\ in\ seconds$$

$$a = semimajor\ axis\ in\ meters$$

$$m_1 = mass\ of\ \ first\ object\ in\ kg$$

$$m_2 = mass\ of\ \ seecond\ object\ in\ kg$$

$$G = Universal\ \ gravity\ constant$$

(See Reference 3)

Mathematical Forumulations

- Newton's Law of Gravitational acceleration:

$$\mathbf{F}_{12} = -G\frac{m_1 m_2}{|\mathbf{r}_{12}|^2}\,\hat{\mathbf{r}}_{12}$$

See Reference 4

- Acceleration due to gravity, rearranging above and plugging in F=ma

$$\vec{a} = -\frac{GM}{r^2}$$

See Reference 5

Testing and Numerical Experiments

- Make use of the internet and previous years' projects as provided on the PHYS 210 website to guide my process and compare my outcome
- Begin by using a smaller number of particles than I would eventually like my program to handle, in order to simplify the beginning stages

Testing and Numerical Experiments

- Will investigate the outcome when the initial parameters are varied, by providing different relative inclinations of the colliding galaxies

Numerical Approach

- One galaxy will be given a velocity towards the other, which will be stationary
- I will control time steps, number of particles, initial velocity
- Utilize Finite Difference Approximation to calculate the vector components of acceleration, position, velocity for all system particles

Timeline

October 21-26: Basic research and beginning design of code

October 29-November 8: Implement code

November 11-15: Test code

November 18-22: Run numerical experiments, analyze data

November 23-29-: Report

November 29: **Submit!**

References

1)http://faculty.etsu.edu/smithbj/collisions/collisions.html

2)http://en.wikipedia.org/wiki/Alar_Toomre

3)http://astronomyonline.org/Science/Orbits.asp

4)http://en.wikipedia.org/wiki/Newton's_law_of_universal_gravitation

5)http://laplace.physics.ubc.ca/210/Proposals-2009/01-ALL.pdf

6)http://laplace.physics.ubc.ca/210/Proposals-2012/L1A-All-Proposals.pdf

7)http://bh0.phas.ubc.ca/210/Doc/term-projects/kdv.pdf

CHAOS IN THE MOTION OF N GRAVITATIONALLY INTERACTING PARTICLES

Physics 210 Term Project Proposal

Clayton Riobo

OVERVIEW

- Chaos theory: a small change in initial conditions produce very different outcomes for some dynamic systems
  - i.e. n-body problem of N=3 or more
- N-body problem predicts the path of particles as they interact gravitationally

PROJECT GOALS

- Write a code in MATLAB that will simulate the paths of n particles as well as trace said paths in two dimensions
- Observe the change in the paths of the particles when initial conditions are changed slightly

MATHEMATICAL FORMULAE

- Gravitational force on the j-th particle

$$F_{ji}(t) = \sum_{i=1}^{N} \frac{G M_j M_i}{r^2{}_{ji}(t)} \boldsymbol{r}_{ji}(t)$$

$$\boldsymbol{r}_{ji}(t) = \frac{\overrightarrow{r_j} - \overrightarrow{r_i}}{\left|\overrightarrow{r_j} - \overrightarrow{r_i}\right|}$$

Numerical Approach

- Finite difference approximation

Visualization and Plotting Tools

- To be learned in future lectures

TESTING AND NUMERICAL EXPERIMENTS

- Vary particles’ mass and initial positions
- Increase N as much as possible

PROJECT TIMELINE

Dates	Activities
10/15-10/26	Basic research, derive equations, begin code design
10/27-11/15	Implement code
11/16-11/19	Test code
11/20-11/26	Numerical experiments, analyze data, begin report
11/27-11/29	Finish report
11/29	Hand in

REFERENCES

http://laplace.physics.ubc.ca/210/Proposals-2012/L1A-All-Proposals.pdf

http://laplace.physics.ubc.ca/210/Proposals-2009/01-ALL.pdf

http://bh0.phas.ubc.ca/210/Doc/term-projects/kdv.pdf

http://en.wikipedia.org/wiki/Chaos_theory

http://en.wikipedia.org/wiki/N-body_problem

QUESTIONS?
COMMENTS?
SUGGESTIONS?
APPLAUSE?

SIMPLE EARTHQUAKE MODEL

PHYS 210 Term Project Proposal

Birgit Rogalla

24 October 2013

OVERVIEW

- Gutenberg-Richter law:

Distribution of earthquake magnitudes for many different fault lines

- Simplified model will be used to determine what properties a system must have to exhibit a power law distribution of earthquake sizes

PROJECT GOALS

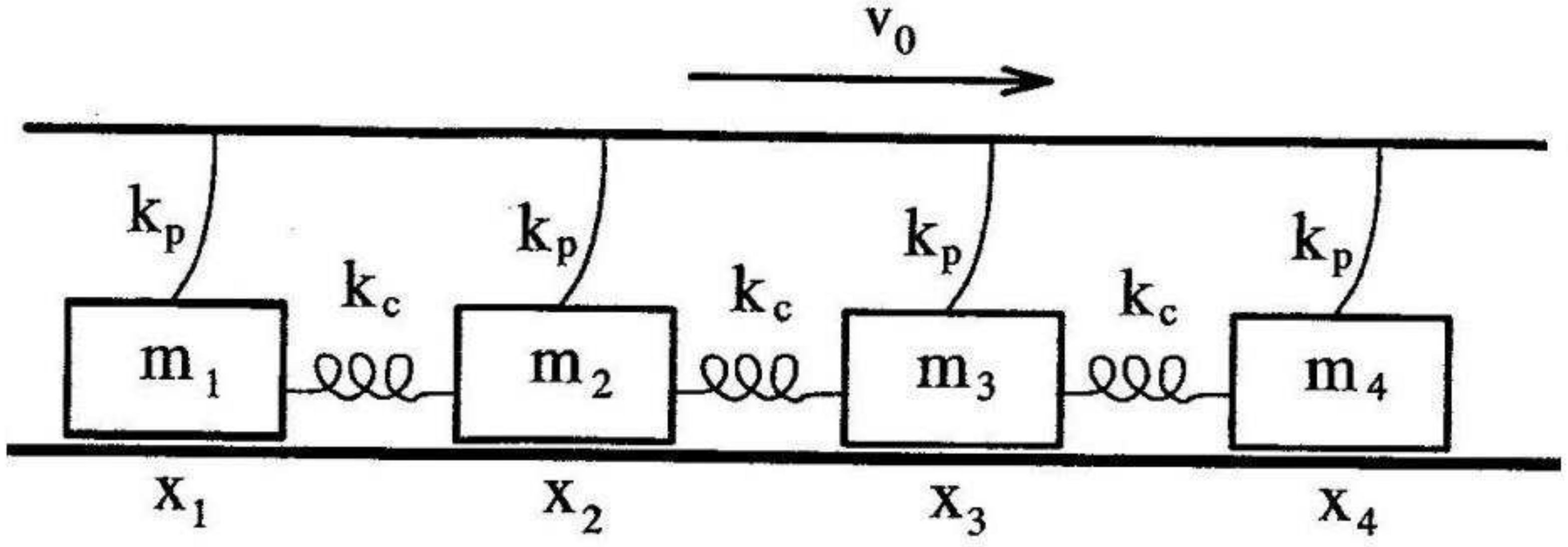
- To write a MATLAB (octave) code which solves the equations of motion used for the Gutenberg-Richter law numerically, using finite difference techniques

- To investigate a range of initial conditions for the equation:
  - Ordered initial configuration – all blocks in equilibrium positions
  - Disordered initial configuration – initial displacement from equilibrium (more realistic)

MATHEMATICAL FORMULATION

Gutenberg-Richter law; Burridge and Knopoff model(image source: Giordano)

- Two plates moving relative to each other



Equation of motion for each block:

$$m_i \frac{d^2 x_i}{dt^2} = k_c(x_{i+1} + x_{i-1} - 2x_i) + k_p(v_0 t \ - x_i) + F_{friction}$$

Rewritten as two differential equations:

$$v_i = \frac{dx_i}{dt}$$

$$m_i \frac{dv_i}{dt} = \ k_c(x_{i+1} + x_{i-1} \ - 2x_i) + k_p(v_0 t \ - \ x_i) + \ F_{friction}$$

Forces on block i from its neighbouring blocks:

$$F_b = -k_c(x_i - x_{i+1}) - k_c(x_i - x_{i-1})$$

Gutenberg-Richter Law:

$P\left(M_{ag}\right) = AM^{-b} = Ae^{-bM_{ag}}$ where $M_{ag} \equiv \ln(M)$

NUMERICAL APPROACH

- Value of parameters for typical behavior:

  m = 1, $\mathrm{k_p} = 40, \mathrm{k_c} = 250, \mathrm{F_0} = 50, \mathrm{v_0} = 0.01$

- Ordered initial configuration – all blocks move together; $\mathrm{v_i} = 0$, all blocks are located in equilibrium positions, $\sum F = 0$ for all springs at t=0
- Time step: velocity (v) used to estimate position at each Δt, forces calculated to obtain v at next time step; two different sizes of Δt :
  - Large Δt when no blocks are moving (only top plate moves with constant v)
  - Small Δt when blocks are moving
- Repeat process but with initial displacement from equilibrium (more realistic)
- Plots of position and velocity vs. time

Back to Gutenberg-Richter Law: $P\left(M_{ag}\right) = AM^{-b} = Ae^{-bM_{ag}}$

$M_{ag} \equiv \ln(M)$ and $M = \sum_{n=time} \sum_{i=blocks} v_i \Delta t$

Sums the total displacement of each block over the course of the event which when combined with the original law results in the magnitude of the earthquake

TESTING & NUMERICAL EXPERIMENTS

- Examine the result of changing parameters by slight amounts
- Will create a number of graphs from model and compare these to the expected behavior

PROJECT TIMELINE

Dates	Activities
10/17-10/25	Research topic, equations & begin code design
10/26-11/15	Implement code
11/16-11/19	Test code
11/20-11/26	Run numerical experiments, analyze data, begin report
11/27-11/29	Finish report
11/29	Submit project

References:

Giordano – *Earthquakes and self organized criticality*

QUESTION?
COMMENTS?
SUGGESTIONS?

Simulaton of N interacting bodies

Or…..

What our universe would look like if it were MUCH Smaller ...assuming that it didn’t fall apart or collapse instantaneously.

Overview

Celestial bodies have always fascinated humans; predicting the interaction between different bodies dates back to the Ancient Maya and Greeks.

Later Newton gave the first clear definition of gravity “all bodies must attract each other”

Bernoulli was the first to solve the two bosy problem, which will be also be my starting point for simulation

Project Goals

- Simulate the N masses in space interacting using a MATLAB code and finite differencing techniques, for equations of motion
- Compare the simulation with “real-world” interactions, in the literature
- Experiment with different initial conditions

Mathematical Formulation

Newton’s universal law of gravitation:

$$F = G\frac{m_1 m_2}{r^2}$$

F=force between the two masses,
m1 and m2 are two masses, r is the distance between the masses
and G is the gravitational constant:6.674×10^{-11} N m^2 kg^{-2}

F=ma Where m is mass and a is acceleration

General planar motion equations:

$$\mathbf{r} = \mathbf{r}\left(r, \theta, t\right) = r\hat{\mathbf{e}}_r$$

$$\mathbf{v} = \hat{\mathbf{e}}_r\frac{\mathrm{d}r}{\mathrm{d}t} + r\omega\hat{\mathbf{e}}_\theta$$

$$\mathbf{a} = \left(\frac{\mathrm{d}^2 r}{\mathrm{d}t^2} - r\omega^2\right)\hat{\mathbf{e}}_r + \left(r\alpha + 2\omega\frac{\mathrm{d}r}{\mathrm{d}t}\right)\hat{\mathbf{e}}_\theta$$

Numerical Approach

- Using the finite differencing models we have been discussing in class I will approximate the equations of motion by solving for position in terms of the first derivative(velocity) and the second derivative(acceleration), as indicated above.
- Error will be approximated using the same methods
- Sample equation that has been using in past solutions, to check my work against:

$$\underset{i}{\overset{2}{x}}_j(t) = G \sum_{k=0,k\neq i}^{n} \frac{\underset{k}{m}\left(\underset{k}{\overset{0}{x}}_j(t) - \underset{i}{\overset{0}{x}}_j(t)\right)}{\left(\left(\underset{k}{\overset{0}{x}}_1(t) - \underset{i}{\overset{0}{x}}_1(t)\right)^2 + \left(\underset{k}{\overset{0}{x}}_2(t) - \underset{i}{\overset{0}{x}}_2(t)\right)^2 + \left(\underset{k}{\overset{0}{x}}_3(t) - \underset{i}{\overset{0}{x}}_3(t)\right)^2\right)^{3/2}}$$

Visualization and Plotting tools

Plotting will be done using MATLAB;

Simulation will be visualized in an mpeg; though at this time I am unfamiliar with how to create the mpeg from the approximations

Testing and Numerical Experiments

As previously noted, we don’t want our mini universe to implode or explode, this will hopefully be avoided

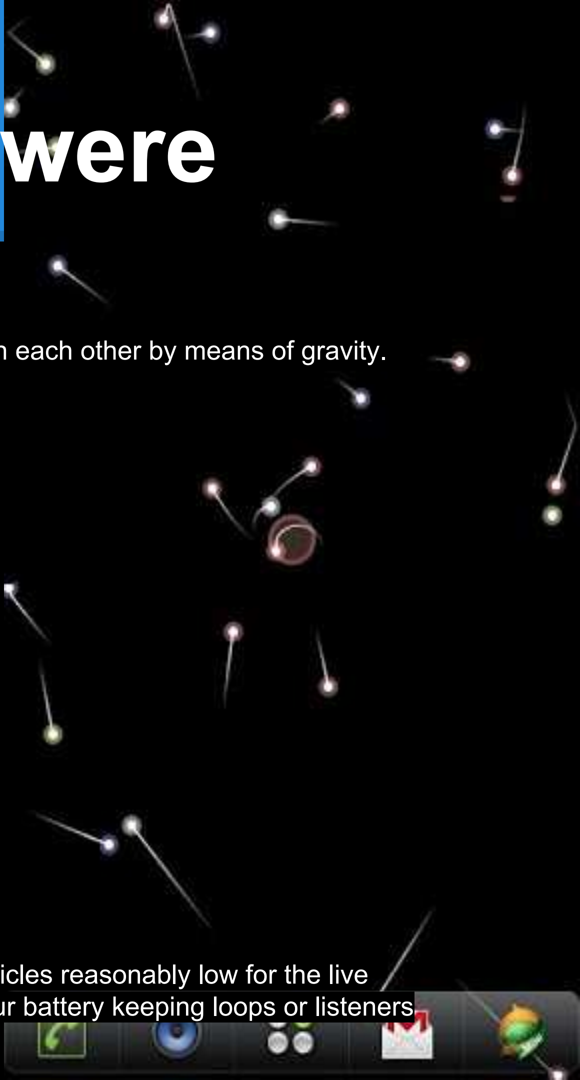
Timeline

- 10/22->10/27 Research project and design code
- 10/27->11/16 Implement code
- 11/16->11/20 Test Code
- 11/20->11/25 Run Experiment and Analyze Data
- 11/25->11/29 Write Report
- 11/29 Celebrate the last day of classes
- 11/30 Finish and submit report

References

- http://en.wikipedia.org/wiki/N-body_problem
- http://en.wikipedia.org/wiki/Newton's_law_of_universal_gravitation
- http://en.wikipedia.org/wiki/Equations_of_motion
- http://www.princeton.edu/~rvdb/WebGL/DoubleDouble.gif
- http://www.appszoom.com/android_themes/wallpapers/nbody-live-wallpaper_ursc.html

What I wish my simulation were



The Live Wallpaper version of the Standalone N-Body Simulator.

Have a real gravitational particle simulation in the background of your phone where particles interact with each other by means of gravity.

Features:

-Real n-body simulation, pure gravitational interactions between particles.

-Allow or not phisically realistic collisions and mergers between particles.

-Set particle and background colors.

-Set gravity strength and particle mass.

-Accelerometer support.

-Shoot particles or create repulsive forces using touchscreen.

-Enable the action to expand beyond your screen limits without losing particles.

-Enable or disable the central black hole which exerts a neat attractive force towards the center.

-Enable or disable particle trails (disable to improve performance).

Take into account we're talking about a real n-body simulation. I recommend keeping the number of particles reasonably low for the live wallpaper version (about 10-20), even though it only executes when the screen is on. Does not drain your battery keeping loops or listeners when sleeping.

Thank You!

Galaxy Collisions

Matthew Russell

2013

Overview

- Using the Toomre model to simulate two galaxies colliding.
- Approximate stars as particles centred around the galaxy core i.e. that's where the bulk of the mass will be.
- The centre of each galaxy will interact with the other until only we have one galaxy.

Goals

- Use MATLAB to create a code that will take the initial conditions and give an output that will simulate the galaxy collision.
- See how the output varies with different initial conditions. i.e. change the initial velocity or mass or number of particles.

Mathematical Formulas

- $\overrightarrow{F_{gi}} = m_i\overrightarrow{a_i} = m_i \frac{d^2\overrightarrow{r_i}}{dt^2} = G \sum \frac{m_i m_j}{{r_{ji}}^3}\overrightarrow{r_{ji}}$ where
  $\overrightarrow{r_{ji}} = \overrightarrow{r_i} - \overrightarrow{r_j}$

- Might need Poisson's equation:
- $\nabla^2\Phi(x,t) = 4\pi G\rho(x,t)$
- Get to latter
- Centre of mass:
- $\vec{R} = \frac{1}{M}\sum_{i=1}^{n} m_i\vec{r_i}$

Numerical approach 1

- The simplest way to do this is calculate $\vec{F}$ of each particles on other and on opposite galaxy. Can be simplified by having no collisions and approximate that particle mass will be much smaller than galaxy mass.
- Use FDA. Specify domain of solution on independent variables. Create a 2D mesh which consists of a finite set of grid points.

Numerical approach 2

- Use a tree system such as the Barnes-Hut tree or algorithm.
- This divides n bodies into groups and stores them in a quad tree(Tree structure that will continuously divide the space into quadrants)
- The top is the whole space, then one down is split into four and so on.
- The net force on a body is calculated using centre of mass. Start at the top level and if other particles(which will be in separate quadrants and levels within the tree) are far enough then can approximate a centre of mass, if not go a level done and try again.

Numerical approach 3

- The particle mesh. The particles are taken and allocated into grid based on how dense they are.
- The Poisson potential is then solved using FDA for this grid and forces are applied to each particle dependent on that particles location within the grid with appropriate invariants and domain.

Timeline

- By end of October know I'm going to do galaxy collisions with what numerical approach and how that approach will work, i.e. know all the maths of it FDA's etc.
- Up to the 18th of November write and test code
- By 25th November apply numerical experiments and analyse data.
- For the due date write up report and final presentation.

References

- http://en.wikipedia.org/wiki/Barnes%E2%80%93Hut_simulation
- http://en.wikipedia.org/wiki/N-body_simulation
- http://en.wikipedia.org/wiki/Particle_Mesh

MODELING A CHARGED PENDULUM SWINGING OVER AN ARRAY OF DISCREET CHARGES

Physics 210 Term Project Proposal

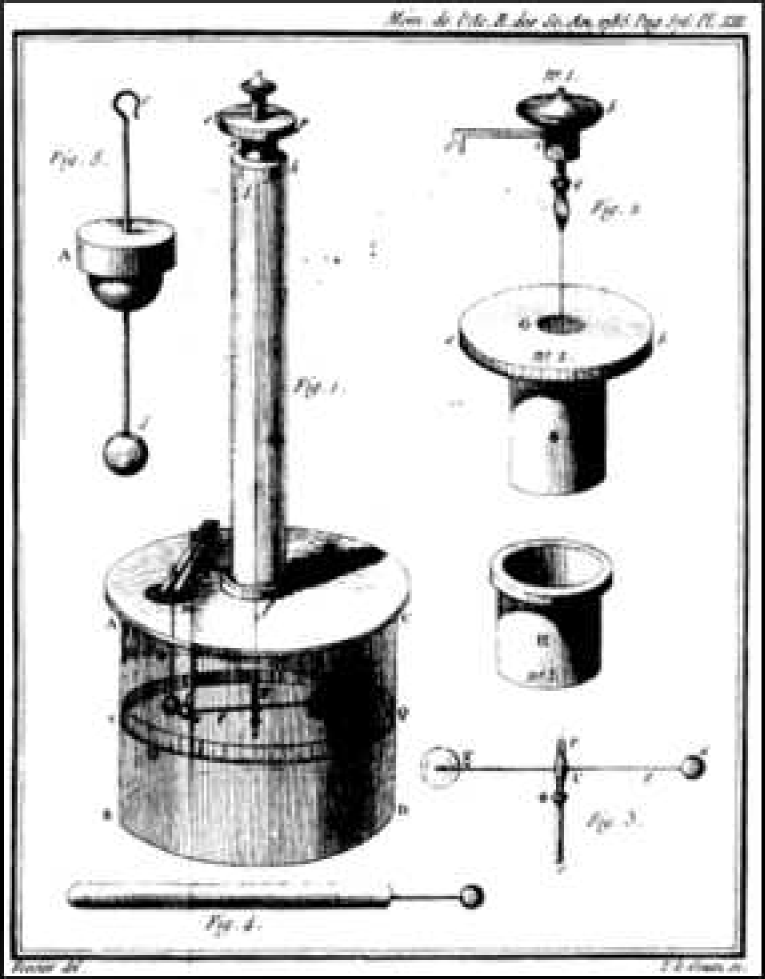
Devin Thomas

OVERVIEW

- A pendulum swinging in three dimensions
- The pendulum is charged and has some mass
- It is swinging above a board on which point charges are held in place
- The pendulum can be considered to be confined to a sphere
- It's motion can be characterized by ODE's which can be approximated by finite difference

PROJECT GOALS

- Use matlab/octave code to model the pendulum
- Render's based upon polar coordinates Θ and r.
- Allow for runtime decision of parameter's , including non constant charges/charge positions if feasible.
- Examine correctness of solution and model using convergence tests with known solutions
- Investigate a variety of initial conditions, symmetrical and asymmetrical, and if feasible a variety of changing conditions.

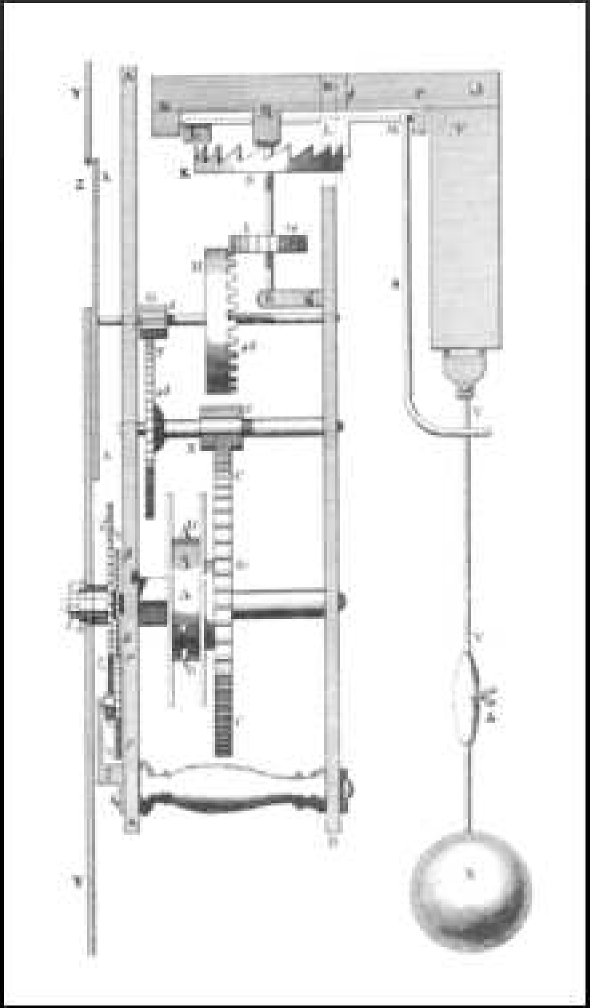


MATHEMATICAL FORMULATIONS

- $\mathcal{T}=(mr^2)\alpha=F\times r$
- Force is sum of gravitational and coulomb forces
- $\Theta''=F\times r/(mr^2)$
- Use finite difference to step Θ along from Θ''

NUMERICAL APPROACH

- The magnitude of the angular displacement will be found using the finite difference approximation from the angular acceleration.
- The direction will be the normalized vector of Fxr
- F will be the three directional vector sum of the gravitational force mg and the coulomb forces $kq\Sigma(Q/r^2 * \hat{r})$ ::This r is not the pendulum's radius it is the displacement from the charges to the bob.
- The mr^2 term will be calculated as the scalar $m|r|^2$ and will be considered constant unless a changing radius pendulum is implemented.



TIMELINE

Dates	Activities
10/15-26	Hammer out equations, complete simple implementation
27-15	Implement data visualization, implement additional features
16-23	Test code, run experiments
24-26	Analyze data, tweak experiments, begin report
27-29	Finish report
Before 12/02	Submit project

QUESTIONS ETC?

- (all the images are from the wikipedia simple pendulum page, the replica seismograph is used under creative commons license, the rest are in the public domain.)

Traffic Simulation via Cellular Automata

By: Austin Wallace

Overview

- I will be using a relatively simple cellular automata model to simulate traffic under various conditions.
- I will then be analyzing that data and examining the traffic flow and density, and finding the ideal conditions to balance the two.

Project Goals

- I will use the Nagel-Schreckenberg model to first simulate the traffic flow of a one-lane road.
- After finding the ideal conditions for a simple one lane road, I will attempt to add other factors and examine how they affect traffic flow. Possible factors include: On/off ramps, cross streets, different maximum speeds, and different probabilities of random braking.

Numerical Approach, Nagel-Schreckenberg Model

- I will be using a one-dimensional array (1-land road) of cells, where each cell will either have a value of 1 (filled/car) or 0 (empty).
- The model defines four basic rules:

1. A car moves moves V cells in one time-step, T.
2. A car moving with velocity "V", will accelerate to velocity "V+1" if it has room, barring rule 4.
3. If, at its current velocity, the car will hit another car, it will instantaneously decelerate so that it doesn't hit the car.
4. Even if it can go faster, every car has a probability of decelerating or staying at constant velocity at every time-step.

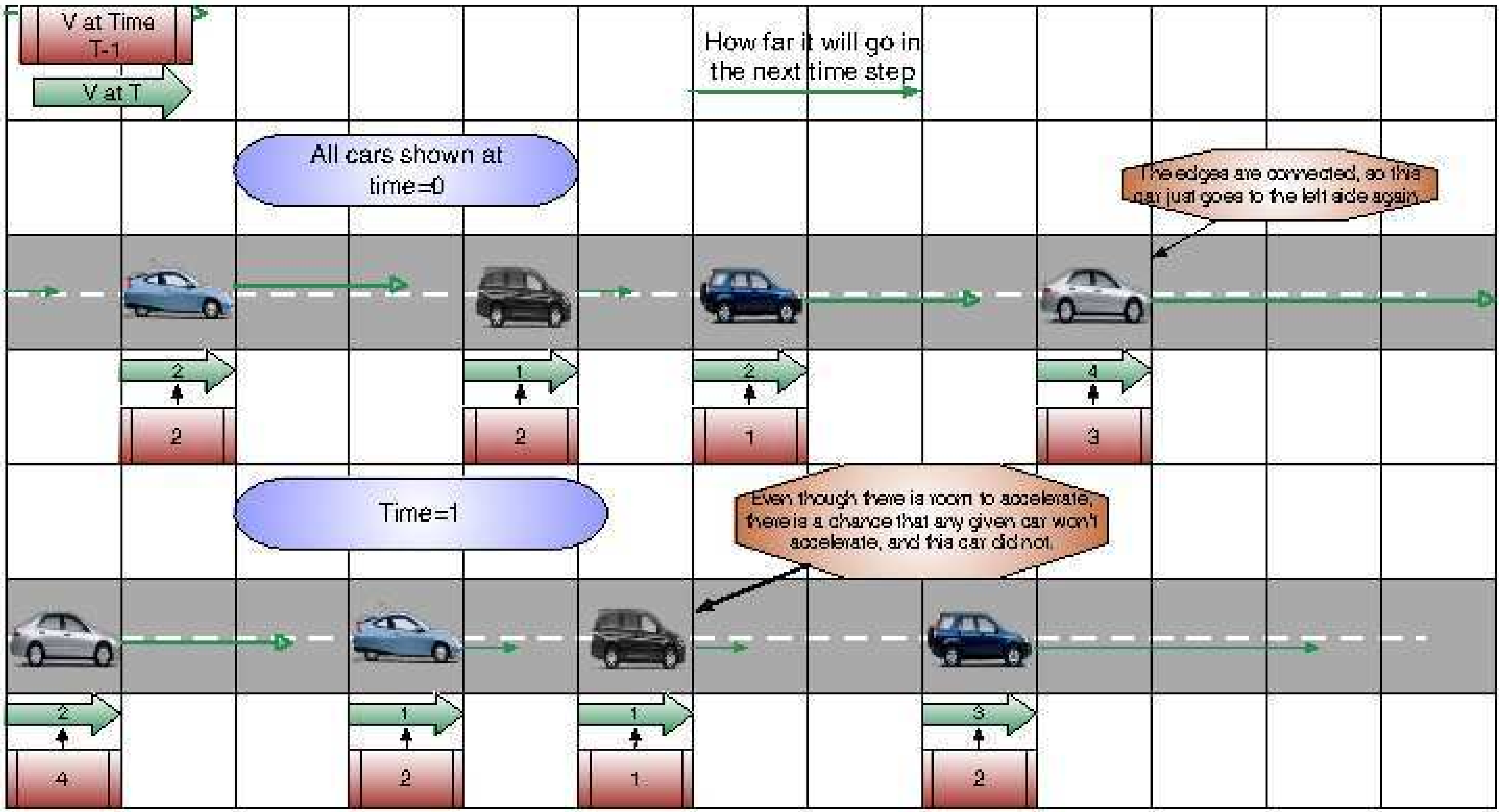
Also, I will be using a periodic edge-case, so a car that goes off the screen to the right will appear on the left-edge of the screen.

Visualization and Plotting

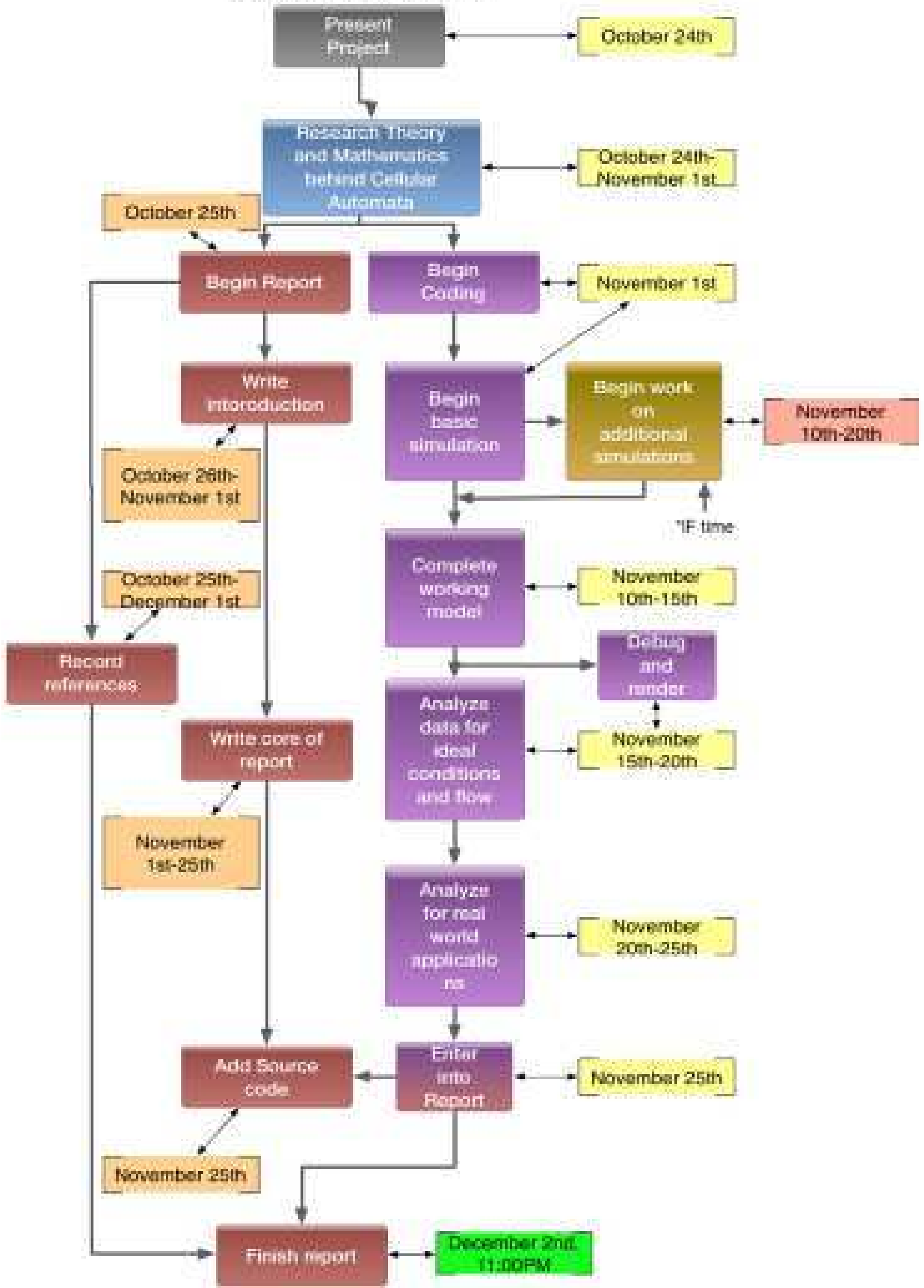
- I will be using Matlab to code my simulation and rendering tools to visualize it.

The next page contains a rendering of how the initial simulation might look.

Omnigraffle Simulation


V at Time T-1
V at T
How far it will go in the next time step
All cars shown at time=0
The edges are connected, so this car just goes to the left side again
2
2
1
2
2
1
4
3
Time=1
Even though there is room to accelerate, there is a chance that any given car won't accelerate, and this car did not.
2
4
1
2
1
1
3
2

Project Timeline
October 24th-December 2nd



References

- Nagel and Schreckenberg, “A cellular automaton model for freeway traffic”
- http://mathworld.wolfram.com/CellularAutomaton.html
- http://en.wikipedia.org/wiki/Nagel-Schreckenberg_model
- http://natureofcode.com/book/chapter-7-cellular-automata

Finite Difference Approximation of the N-body Problem

PHYS 210 Term Project Proposal

Tianrui Xu

Overview

- Simulation of the future (equilibrium) state of n particles
- No fields being considered except that generated by particles
- No reactions (e.g. collision/decay) involved
- 2-D

Project Goals

- Solve the N-Body Problem in 2-D numerically using Matlab
- Particles with 2 different charges (+1C, neutral)
- Particles with two different positive masses (1kg, 2kg)
- Visualize the solution

Mathematical Formulation

- Gravitational interaction between 2 identical particles

$$F_G = G\frac{m^2}{r^2} \qquad (1)$$

- Coulomb force between 2 charged particles

$$F_q = k\frac{q^2}{r^2} \qquad (2)$$

Mathematical Formulation

- Newton's Second Law

$$\vec{F} = m\vec{a} \qquad (3)$$

- Motion

$$\vec{a} = \frac{\vec{F}_G + \vec{F}_q}{m} \qquad (4)$$

$$\Delta\vec{v} = \int_t^{t+\Delta t} \vec{a}\,dt \qquad (5)$$

$$\Delta\vec{s} = \int_t^{t+\Delta t} \vec{v}\,dt \qquad (6)$$

Mathematical Formulation

- Potential (used for testing)

$$W = -\int_{s}^{\Delta s} \vec{F} \bullet d\vec{s} \qquad (7)$$

$$\Delta U = -W \qquad (8)$$

Numerical Approach

- Generate random particles at the beginning as initial condition
- Randomness:
  - The number of particles
  - The initial velocities and positions of particles
  - The charge and mass of particles (+1C or neutral)

Numerical Approach

- Calculate Gravitational Force and Coulomb Force using Eqn. (1)(2), and thus get acceleration using Eqn.(4), and thus velocity and distance Eqn.(5)(6)

- Use loops to calculate acceleration, velocity and distance at different times. The loop variable is Δt

Visualization and Plotting Tools

- Use xvs for interactive analysis and generation of mpeg animations
- Use Matlab/Gnuplot to plot if necessary

Testing and Numerical Experiments

- Testing
  - Use Eqn.(7)(8) to test that the potential goes to minimum as the system moves to equilibrium.

    (a U-t plot to visualize the tendency)
  - Generate a small number of particles (i.e. 2 or 3 or 4) to check if the result is plausible

- Numerical Experiments
  - Simulate motions of large number of particles in 2-D

Project Timeline

Dates	Activities
20131021-1025	Do Research
20131025-1115	Write and Implement Code
20131116-1119	Test Code
20131120-1126	Run numerical experiments, Do data analysis, Begin report
20131127-1129	Finish report
20131130-1202	Polish and submit report

References

- Halliday et al, Fundamentals of Physics 6th ed.
- http://bh0.phas.ubc.ca/210/Doc/term-projects/kdv.pdf

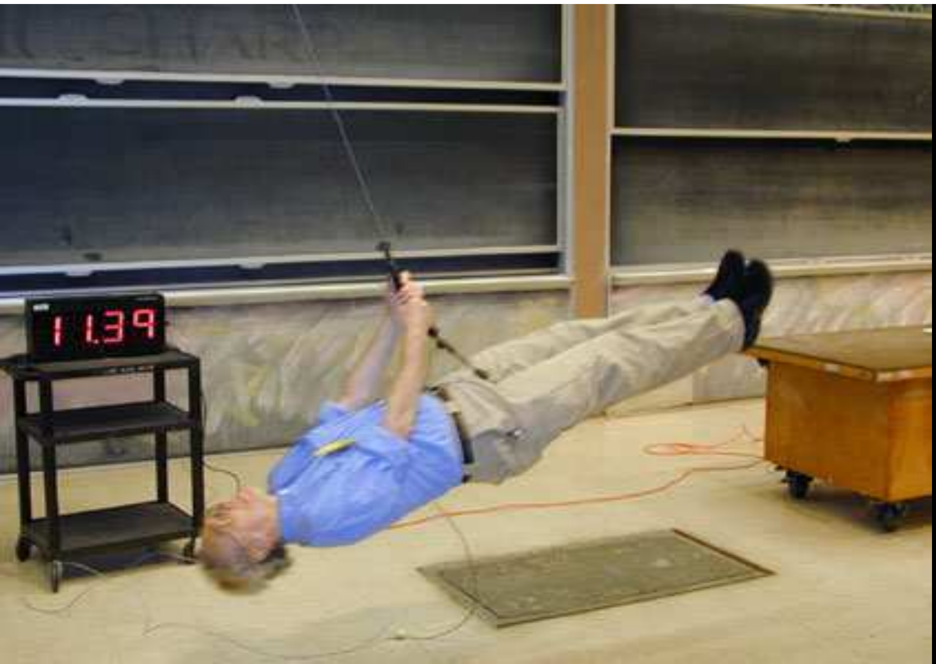
Simulation of the Simple and Compound Pendulums

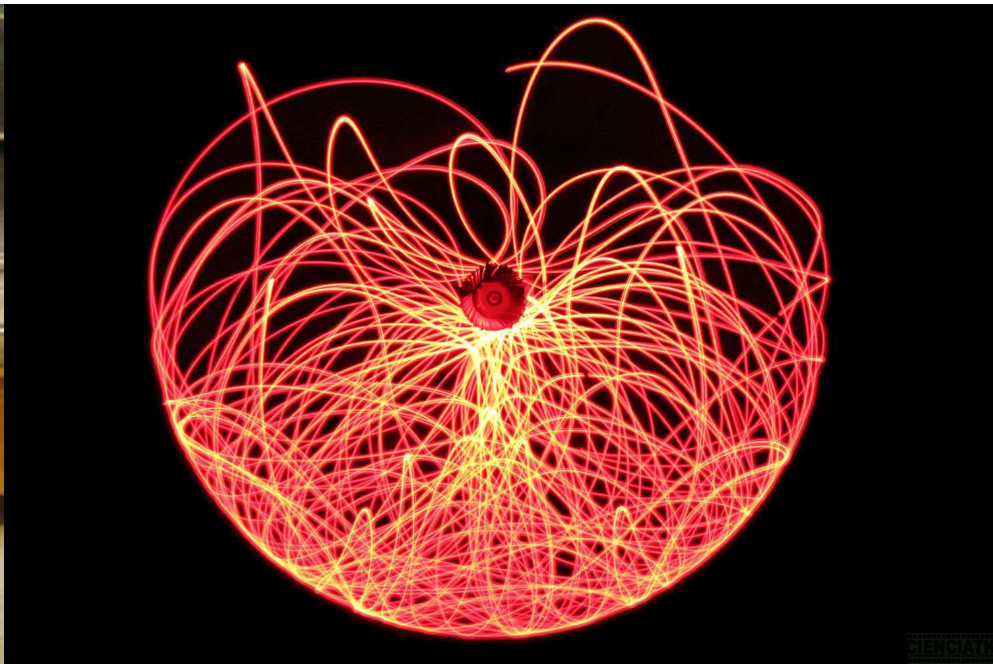
PHYS 210 Term Project Proposal

Tao Zheng

Overview and Goals

- Simple pendulums exhibit elliptic periods at higher equilibrium positions
- Compound pendulums exhibit chaotic motion (no period)





Goals

- To simulate the trajectory of simple and compound pendulums using MATLAB (Octave)

- To test the simulation at various equilibrium positions (release angles)

- To experiment with varying masses and arm lengths

Some Mathematics

- Simple pendulums have periods expressed in elliptic functions

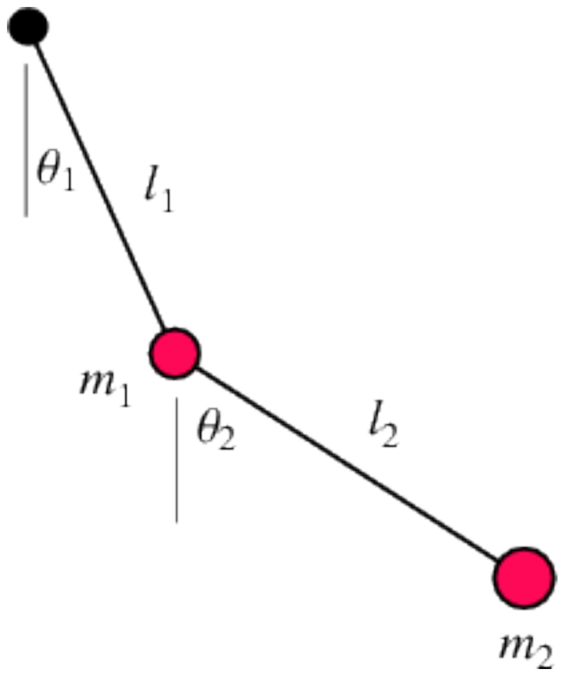
$$T = 2\pi\sqrt{\frac{L}{g}}\left(1 + \frac{1}{16}\theta_0^2 + \frac{11}{3072}\theta_0^4 + \cdots\right)$$

- The motion is predictable until…

Mathematics (cont.)

- We attach pendulums on pendulums

$$\begin{aligned} x_1 &= l_1 \sin\theta_1 \\ y_1 &= -l_1 \cos\theta_1 \\ x_2 &= l_1 \sin\theta_1 + l_2 \sin\theta_2 \\ y_2 &= -l_1 \cos\theta_1 - l_2 \cos\theta_2. \end{aligned}$$

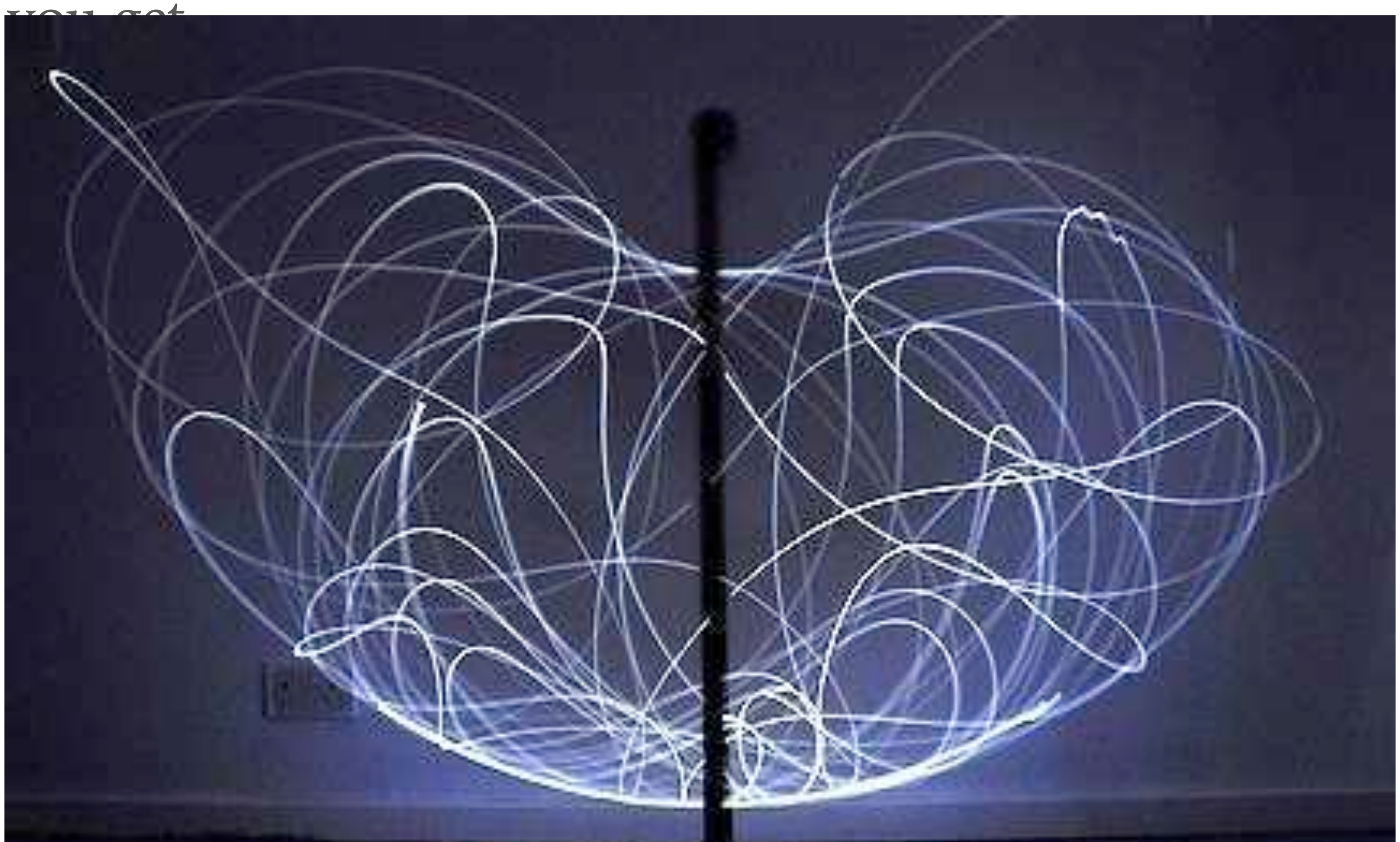


- The Lagrangian of the system is:

$$T - V = \frac{1}{2}(m_1 + m_2) l_1^2 \dot{\theta}_1^2 + \frac{1}{2} m_2 l_2^2 \dot{\theta}_2^2 + m_2 l_1 l_2 \dot{\theta}_1 \dot{\theta}_2 \cos(\theta_1 - \theta_2) + (m_1 + m_2) g l_1 \cos\theta_1 + m_2 g l_2 \cos\theta_2$$

Mathematical Result

- Solve for the time-dependent angles in the Lagrangian and you get



Numerical Approach

- Finite difference methods
- Possibly implement the Runge-Kutta Algorithm for solving the equations of motion derived from the Lagrangian

Visualization and Plotting

- I will use mpeg animations and MATLAB plotting

- The simulation will be in the xy-plane

- The plot will contain tracings of the trajectories for each mass

Testing and Numerical Experiments

- Test systems with multiple masses (ideally 2 - 3)
- Include adjustable parameters (mass, length, release angle) for each system

Project Timeline

Dates	Activities
OCT 19 – OCT 31	Research, derive equations, begin code design
NOV 1 – NOV 15	Implement code
NOV 16 – NOV 20	Test code
NOV 20 – NOV 26	Run experiments, analyze data, begin report
NOV 27 – NOV 30	Finish report
DEC 1	Submit report

References

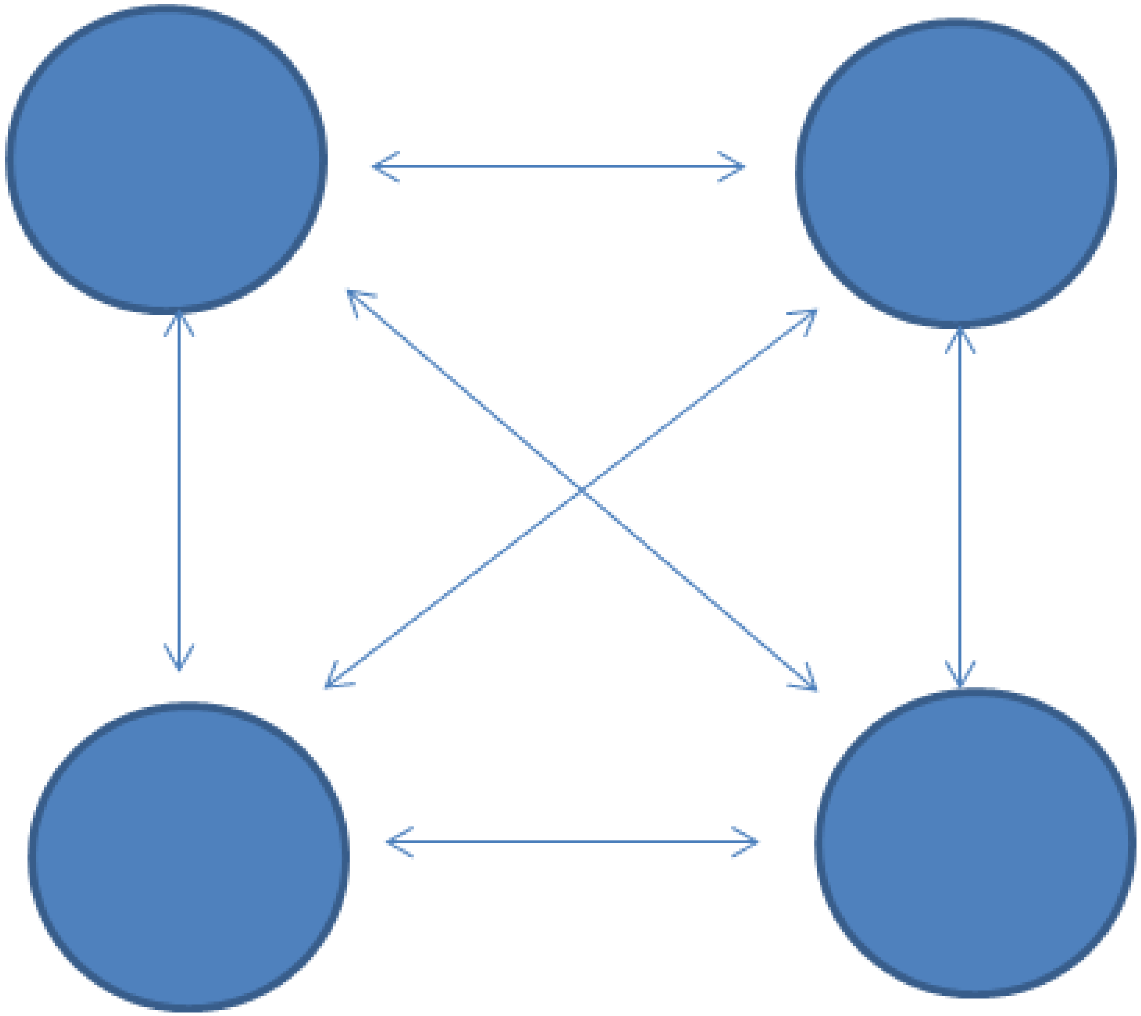
- http://scienceworld.wolfram.com/physics/DoublePendulum.html
- http://en.wikipedia.org/wiki/Pendulum
- http://en.wikipedia.org/wiki/Double_pendulum
- http://www.cienciatk.csic.es/Fotografias/EL+PENDULO+CAOTICO_25755.html
- http://ocw.mit.edu/courses/physics/8-01-physics-i-classical-mechanics-fall-1999/video-lectures/

N-body simulation: testing the stability of a gravitationally linked orbital station (GLOS) using finite difference approximation

PHYS 210 Term Project Proposal

Dhaneesh Kumar

- **Overview**
  - A GLOS is simply a collection of smaller stations that are kept in stable positions relative to each other via gravity only (inspired from phys 107/108)
  - As part of any design process, it is necessary to test the stability of such a setup
- **Project goals**
  - To write a Matlab (octave) code that can simulate a variety of stations under different initial conditions using finite difference approximations
  - To investigate a variety of initial conditions and observe the evolution of the system i.e. observe the stability of the GLOB
  - Tabulate data concerning the different initial conditions to extract any trends or useful information
  - Test the accuracy of the simulation by convergence testing or by repeating the simulation using euler's method



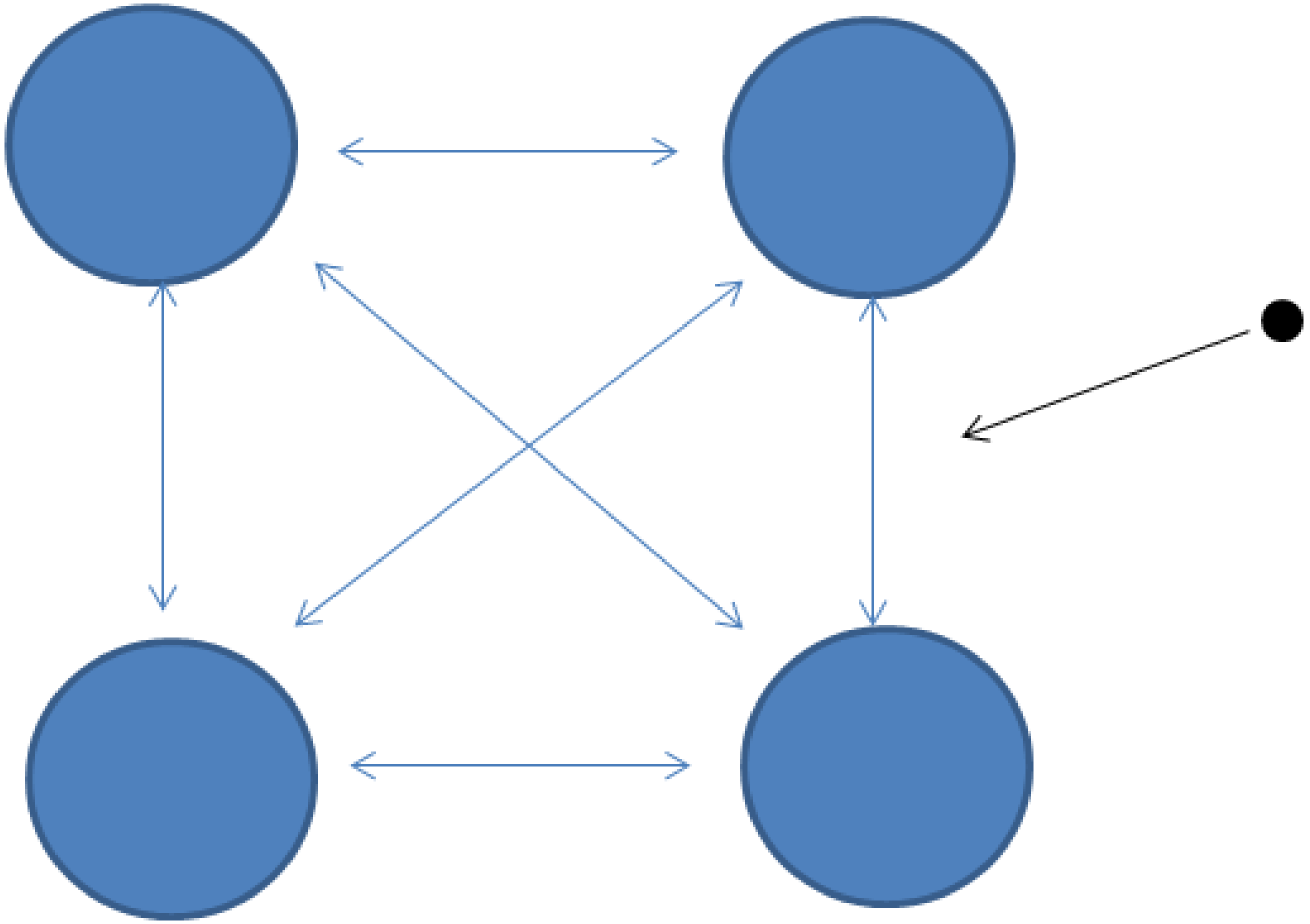
Simple 4-body GLOS

- **Equations that would be used**
  - Newton's Law of Gravity

$$\ddot{\mathbf{r}}_i = -G \sum_{j=1;\, j\neq i}^{N} \frac{m_j(\mathbf{r}_i - \mathbf{r}_j)}{|\mathbf{r}_i - \mathbf{r}_j|^3} .$$

  - Equations of motion

- Experimentation
  - Stability of GLOS would be measured/obsered as the reaction of the GLOS to incoming particles
  - The simulation would be set up using a space mesh in 2D These incoming particles would vary in terms of
    - Speed
    - Mass
    - Direction of Travel
    - Number of particles*
  - The GLOS would be varied in its setup as well



- Testing accuracy of simulation
  - Convergence testing: repeat a particular simulation (i.e. with fixed initial conditions) using smaller discretization scales and observe behaviour. A convergence behaviour should be obtained.
  - Compare simulation with simulation using other methods such as Euler’s method*

- Project Timeline

Dates	Activities
10/18-10/27	Research and basic code design
10/28-11/15	Implement Code
11/16 – 11/20	Test Code
11/21 -11/27	Run Experiments, Analyze data, begin report
11/28-11/30	Finish report
11/30	Submit Report

QUESTIONS?