

2-D Cellular Automata Forest Fire Modeling

Gabriel Aaron

Overview

- ♦ Wildfire has a huge effect on British Columbia's natural environment
- ♦ Computer programs play an essential role in how the six provincial fire centres decide to action fires
- ♦ 2-D cellular automata modeling is an effective tool for a basic simulation of how forest fires spread

Goals

- To write a Matlab program that models the spread of a forest fire using 2-D cellular automata
- To investigate how natural environmental factors such as wind, precipitation and “spotting” influence my basic model
- To compare my model to other ones that can be found online

Mathematical Formulation

- The initial state will consist of a number of coloured cells representing trees
- A tree will have a probability $p(l)$ of being struck by lightning to start a burn
- In iterative time steps, every cell will update. The state of a given tree at time $t+1$ will depend on its current state as well as its neighbouring trees at time t

$$S_{ij}(t+1) \sim S_{i+a, j+b}(t)$$

Where $a, b = -1, 0, 1$ and $a=b=0$ not allowed

	$i-1, j+1$	$i, j+1$		
	$i-1, j$	i, j	$i+1, j$	
		$i, j-1$		

Probability that cell a burns

$$P(a_b)_{t+1} = (1-p^{N(b)})_t$$

$$0 < p < 1$$

Where $N(b)$ is the number of burning cells
neighbouring cell a

Testing

Run trials with my basic model to ensure an “acceptable” amount of realism in the simulation

Compare to other models found online

Implement environmental effects through the altering of probability values (for example, wind would increase the probability of adjacent trees igniting)

Implement spotting effects to allow subsequent ignitions to occur at cells outside the immediate neighbourhood of a burning cell

Timeline

October 22 - October 31	Research and get accustomed to necessary software
November 1 - November 5	Design code
November 6 - November 15	Write code and implement visual component
November 16 - November 19	Test with various parameters
November 20 - December 1	Analysis/Writeup/Submission

References

- Ventosa, Ignasi. "CELLULAR AUTOMATA MODELS. A USEFUL MODELLING TOOL TO DEFINE ENVIRONMENTAL POLICY: THE CASE OF THE FOREST FIRES." Dept. of Geography King's College London United Kingdom. Web. 20 Oct. 2014. <<http://www.fundacioent.cat/images/stories/ENT/articles/automata.pdf>>
- .Quartieri, Joseph, Nikos Mastorakis, Gerardo Iannone, and Claudio Guarnaccia. "A Cellular Automata Model for Fire Spreading Prediction." Web. 20 Oct. 2014. <<http://www.wseas.us/e-library/conferences/2010/Corfu/UPT/UPT-26.pdf>>

Simple Earthquake Model

Jonah Berean

PHYS 210 Term Project Proposal

Overview

Gutenberg-Richter Law:

Amount of relative plate shifting - proportional to the moment of the event and the energy released.

Will develop a simplified model to determine effect of system properties on magnitude.

Project Goals

- To write MATLAB code that uses FDA to solve equations of motion for the Gutenberg-Richter Law.
- To investigate varied initial conditions of the plate (block) arrangement.

Mathematical Formulation

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

Two plates moving relative to one another

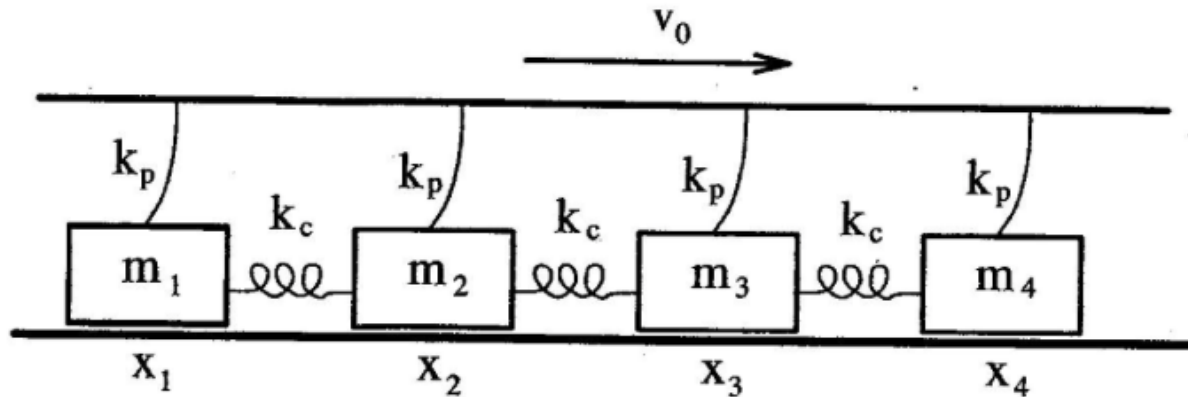


FIGURE 12.14: Model of two plates separated by a fault line at which an earthquake can occur. Imagine this to be a top view of Earth's surface, with the fault line running between the blocks and the bottom plate.

Gutenberg-Richter Law:

$$P(\mathcal{M}) = A M^{-b} = A e^{-b\mathcal{M}} . \quad (12.2)$$

Equation of motion for individual block:

$$m_i \frac{d^2 x_i}{dt^2} = k_c (x_{i+1} + x_{i-1} - 2x_i) + k_p (v_0 t - x_i) + F_f . \quad (12.9)$$

Can be rewritten as two first order differential equations:

$$\frac{dx_i}{dt} = v_i , \quad (12.10)$$

$$m_i \frac{dv_i}{dt} = k_c (x_{i+1} + x_{i-1} - 2x_i) + k_p (v_0 t - x_i) + F_f , \quad (12.11)$$

Numerical Approach

Equations of motion will be discretized using a first order FDA.

Two time steps will be used in order to deal with problem of disparity between

- time between quakes (large step when blocks are not moving)

- time steps within quake activity (small step when blocks are moving)

Position estimation will be done using velocities at each time step (calculated using forces)

Plotting of position and velocity versus time

Summing the total displacements of the plates will yield the magnitude of the earthquake, once combined with original Gutenberg-Richter Law.

$$P(M_{ag}) = AM^{-b} = Ae^{-bM_{ag}}$$

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

Testing

Examine the results by varying the initial parameters (block position)

Graph the modelled behaviour and compare to expected results.


Project Timeline

Date	Work
October 20th - 31st	Further research, begin coding
November 1st - 15th	Code Implementation and Testing
November 16th - 20th	Numerical analysis, begin project report
November 21st - December 2nd	Finish Project Report
December 3rd	Submit Project

Reference: Giordano - Earthquakes and self organized criticality

**Thank you for your time
and attention.**

Questions?
Comments?
Suggestions?



Ising Model Simulation of a Ferromagnetic Material in an External Magnetic Field

PHYS 210 TERM PROJECT – CONNOR BEVINGTON

Overview

- ▶ Ferromagnetism is described as a collection of atomic spins that align in the same direction, causing a net magnetic moment that can be detected
- ▶ This alignment and subsequent net magnetization occurs when ferromagnetic materials are subjected to an external magnetic field, causing the randomly-oriented spins to adopt a parallel orientation
- ▶ Above a critical temperature, known as the Curie temperature, these materials lose their magnetism due to thermal motion outcompeting dipole alignment
- ▶ Ferromagnetic materials include nickel, iron, cobalt, and some rare earth metals, including many of their alloys.
- ▶ We use ferromagnetic materials on an everyday basis; their applications seen in electric motors, step-up and step-down transformers, and hard disk storage

Project Goals

- ▶ To write a MATLAB code that models the net magnetization of a ferromagnetic material using a 50×50 array to simulate individual spin orientations in an external magnetic field, while increasing temperature (and thus the energy of the system) until the Currie temperature is reached
- ▶ Explore two initial conditions/approximations:
 - ▶ The mean-field approximation, where all atomic spins are identical initially
 - ▶ The Monte Carlo method, which steps through each atom in the array and uses an algorithm to determine whether or not to flip the spin
- ▶ Compare the results with known numerical solutions to the non-analytical Ising Model

Physical/Mathematical Formulation

- ▶ Each atom in the array may have a spin state, s_i , either spin up (+1) or spin down (-1)
- ▶ The energy of the system is given by

$$E = -\sum_{\langle ij \rangle} s_i s_j - \mu H \sum_{i=1}^N s_i \quad (1)$$

where $\langle ij \rangle$ refers to summing over the neighbouring atoms and μ is the magnetic moment of the atom

- ▶ So for the i^{th} atom, the energy is given by

$$e_i = -\frac{1}{2} \sum_{k=1}^z s_k s_i - \mu H s_i \quad (2)$$

where we sum to the z closest neighbouring atoms. The $\frac{1}{2}$ factor arises to avoid double counting each pair of neighbours

Physical/Mathematical Formulation (con't)

- ▶ For a given atom in the lattice, the Boltzmann distribution gives us the mean spin of the atom in terms of the Curie temperature, T_c , and the critical magnetic field, H_c

$$\bar{s} = \tanh\left(\frac{T_c}{T} \left(\frac{H}{H_c} + \bar{s}\right)\right) \quad (3)$$

- ▶ The Curie temperature is derived from properties of the material, and the critical magnetic field is given by

$$H_c = \frac{kT_c}{\mu} \quad (4)$$

where k is the Boltzmann constant

- ▶ Finally, we wish to plot the net magnetization, which is defined as

$$M = \mu N \bar{s} \quad (5)$$

where N is the number of atoms (elements) in the array

Numerical Testing/Experiments

- ▶ We can iterate equation (3) for a given mean spin, \bar{s}_i

$$\bar{s}_{i+1} = \tanh\left(\frac{T_c}{T} \left(\frac{H}{H_c} + \bar{s}_i\right)\right) \quad (6)$$

- ▶ As mentioned, in the mean field approximation we assume $\bar{s}_i = \bar{s}$, then use equation (6) to iterate until $\bar{s}_{i+1} \rightarrow \bar{s}_i$
- ▶ We can plot the net magnetization using equation (5), as well as the net energy and the heat capacity of the system, to compare to known numerical models

$$E = -NkT_c \left(\frac{H}{H_c} + \bar{s}\right) \bar{s} \quad (7)$$

$$C = \frac{dE}{dT} \quad (8)$$

Numerical Testing/Experiments (con't)

- ▶ A more detailed analysis comes from the Monte Carlo method, where the each atom in the array is subjected to the following algorithm:
 - ▶ The change in energy of the system, ΔE , is calculated from equation (1) when the spin of the i^{th} atom is flipped (i.e. from +1 to -1, or vice versa)
 - ▶ If $\Delta E < 0$, the spin is flipped
 - ▶ If $\Delta E > 0$, the spin is flipped with probability $P = e^{-\beta\Delta E}$, where $\beta = \frac{1}{kT}$
- ▶ The process is continued until thermal equilibrium is reached. We define this as when the system has equal likelihood of changing from energy state A to energy state B as well as the reverse operation. That is

$$P_a P_{a \rightarrow b} = P_b P_{b \rightarrow a} \quad (9)$$

Numerical Testing/Experiments (con't)

- ▶ Using MATLAB's plotting facility, the 50 x 50 array can be arranged with each element either white (for spin +1) or black (for spin -1), showing the interactions of the atomic spins as the algorithm progresses
- ▶ Again, plots for net magnetization and the energy and heat capacity of the system will be made and compared to known numerical models
- ▶ In any case, the external magnetic field will cause the atomic spins to spontaneously align, before being unable to align once the Curie temperature is reached. At this point, the energy of the system should reach zero, and the heat capacity should spontaneously drop to zero

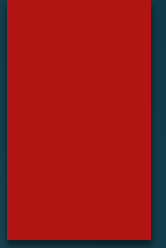
Project Timeline

- ▶ October 21st – 28th
Gather research, formulate a plan for code design
- ▶ October 29th – November 11th
Implement code
- ▶ November 12th – 16th
Test and debug code
- ▶ November 17th – 24th
Run code and conduct numerical analysis
- ▶ November 25th – 28th
Write report and submit, have a celebratory drink

References

- ▶ Fitzpatrick, R. (2006, March 29). The Ising model. *University of Texas at Austin*. Retrieved October 17, 2014, from <http://farside.ph.utexas.edu/teaching/329/lectures/node110.html>
- ▶ Nave, C. R. (2012). Ferromagnetism. *HyperPhysics*. Retrieved October 17, 2014, from <http://hyperphysics.phy-astr.gsu.edu/hbase/solids/ferro.html>

Any Questions?



Toomre Model of Galaxy

Collisions

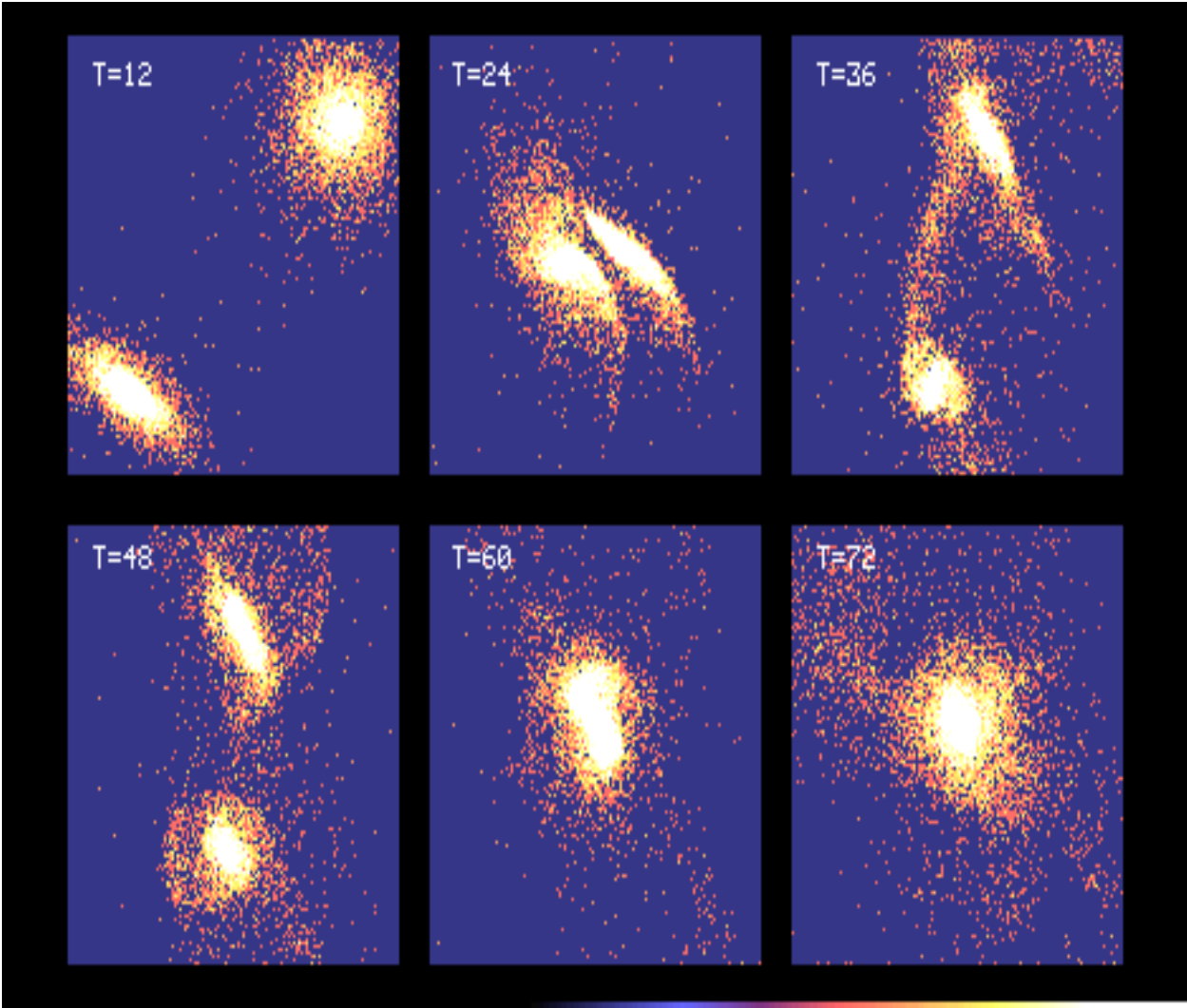
Physics 210 Project Proposal

Joseph Chang

October 21, 2014

Project Overview

- Gravitational interactions between two galaxies is the result of galactic collisions
- In the Toomre model, each star is assigned a mass and sends two digital galaxies moving towards each other
- Using the N-modeling technique, each particle can be used to represent stars and thus further calculate the net gravitational acceleration acting on each particle
- Only stars and galactic cores are represented in this model.



Objectives

- Use Matlab and write a code to show the collisions between two galaxies
- Depict during a galactic cannibalism how small galaxies disrupt and distort the shape of main galaxies
- Create visual representation by generating mpeg animations

Mathematical Formulations

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

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

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

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

Numerical Approach, Experiment's and Testing

- Set initial condition for each galaxy such as the radius, mass, position and velocity
- Stars move initially in circular orbits with mass, position and velocity forming 3-dimensional vectors
- Vary conditions for each galaxy and stars
- Increase N gradually

Project Timeline

Dates	Activities
10/22~10/29	Do basic research, derive equations & begin code designing
10/30~11/05	Write testing script and code
11/06~11/12	Trial and error, test code
11/13~11/30	Run numerical experiment, finish presentation and report
11/30	Submit report (Due date 12/03)

References

1. "Toomre and the first model"
<http://sciencenotes.ucsc.edu/9701/full/features/galaxy/Toomre.html>
2. "Dynamics of Interacting Galaxies"
<http://burro.cwru.edu/JavaLab/GalCrashWeb/dynamic.html>

Diffusion-Limited Aggregation

PHYS 210 Term Project Proposal

Chin (Jean) Chen

Overview



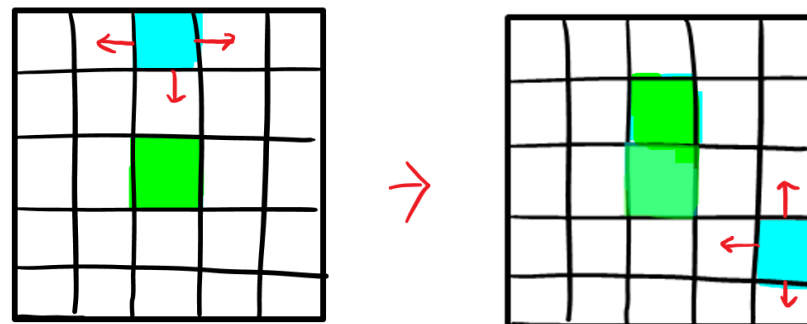
Image credit: Kevin R Johnson (2006)

- Particles diffusing through a fluid:
 - “random walk” (equal probability of moving in any direction)
 - Collide with atoms in the medium (Brownian motion)
 - Hit each other and stick together (or bounce off) and form a cluster called a “Brownian tree”; looks like a fractal
- This process is called “dendritic growth” (structure branches out like trees); modeled by **diffusion-limited aggregation**
- DLA structures in nature:
 - Crystal/Snowflake formation
 - Dielectric breakdown (lightning)
 - Electrodeposition
 - Bacterial colony growth under stress

Algorithm

Simplest case:

- Start with a grid of square cells
- Place a “seed”: a stationary particle in the centre of grid
- Place another particle at a random place and let it diffuse—roam randomly—until it touches the seed and sticks to it
- Place new particle at a random place and let it roam free
- Repeat n times



Project Goals

- Write a MATLAB code to simulate DLA in 2D (3D) with varying initial conditions
- Check how well the model works by comparing with results from other DLA simulations
- Revise code so that particles move according to Brownian motion and move off-lattice
- Link to real-world process—model something like snowflake formation—and plot the clusters via MATLAB, then analyze the structures and compare them to the structures in real-life

Testing and Numerical Experiments


- Vary initial conditions to investigate their effects (e.g. seed position, grid size, total number of repeats, etc.)
- Add some new parameters and observe their effects (e.g. sticking coefficient, changing the number of particles moving around at any point in time; the moving particles can stick with each other and form clusters that diffuse through medium)

Project Timeline

Dates	Activities
10/19-10/27	Basic research, derive equations, begin code design
10/28-11/18	Implement code
11/19-11/22	Test code
11/23-11/28	Run numerical experiments, analyze data, begin report
11/29-12/02	Work on report
12/03 at midnight	Submit project and celebrate

References

- <http://classes.yale.edu/fractals/panorama/physics/dla/dla.html>
- <http://www.tandfonline.com.ezproxy.library.ubc.ca/doi/pdf/10.1080/001075100409698>
- <http://math.rice.edu/~lanius/fractals/dim.html>
- <http://journals.aps.org/prl/abstract/10.1103/PhysRevLett.47.1400>



Simulation of motion of N interacting particles under gravitational forces in 2D plane

TERM PROJECT PROPOSAL

PHYS 210

TONG CHEN (14200142)

Overview

- ▶ The motion of particles in the universe is mainly depends on the gravitational force between them. Gravitational N-body simulations, the numerical solutions of the equations of motions for N particles interacting gravitationally, are widely used tools in astrophysics in the applications from few body or solar system like systems all the way up to galactic and cosmological scales.

Project Goals

- ▶ To solve a 3-body problem with different mass using Matlab programming.
- ▶ To visualize the 3-body motion within a MPEG file.

Mathematical Formulation

- ▶ Newton's universal law of gravitation:

$$F = GM_1M_2/r^2$$

Where F is the gravitational force, G is the gravitational constant (6.67×10^{-11} Newton m^2/kg^2), M_1 is the mass of particle 1, M_2 is the mass of particle 2 and r is the distance between two particles.

- ▶ Kepler's 3rd law:

$$p^2 = 4\pi^2r^3/G(M_1 + M_2) \quad (p^2 = d^3)$$

Where p is the period of the planet orbit, G is the gravitational constant (6.67×10^{-11} Newton m^2/kg^2), M_1 is the mass of particle 1, M_2 is the mass of particle 2, r is the distance between two particles and d is the length of the semi major axis of the ellipse orbit.

Numerical Approach

- ▶ N-body problem can be solved by using finite difference approximations (FDA) to calculate the particle's position and velocity, we can also use FDA to find the force and acceleration of the particle.

Visualization and Plotting Tools

- ▶ All the plotting graph and the MPEG will be outputted using Matlab.

Testing and Numerical Experiments

- ▶ All three particles will be given different mass randomly (no negative and zero mass).
- ▶ Velocities of all particles will be zero at time = 0.
- ▶ All particles will have different initial position randomly.

Project Timeline

Dates	Activities
10/19 – 10/27	Basic knowledge research & code design
10/28 – 11/17	Implement code
11/18 – 11/21	Testing
11/22 – 11/28	Run numerical experiments, analyze data. Begin report
11/29 – 12/01	Finish report
12/02	Project submission

References

- ▶ <http://www.rsmas.miami.edu/personal/miskandarani/Courses/MSC321/lectfiniteDifference.pdf>
- ▶ [http://www.scholarpedia.org/article/N-body_simulations_\(gravitational\)#Introduction](http://www.scholarpedia.org/article/N-body_simulations_(gravitational)#Introduction)
- ▶ http://en.wikipedia.org/wiki/N-body_simulation



QUESTIONS?

N-BODY SIMULATION

Term Project Proposal

Jordan Cooper

What is an N-Body Simulation

- The N-body simulation is a method in which the movement of particles can be calculated
- The particles in the simulation will interact with each other through some fundamental force ie coulomb or gravity

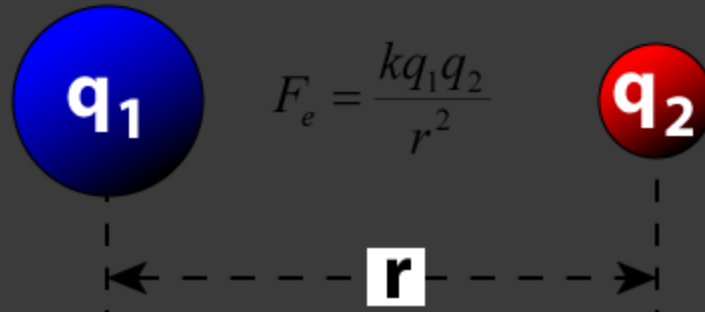
Project Goals

- Write a matlab code which accurately predicts the motion of N particles using the velocity-verlet method
- Run the simulation of N -body particles
- Finish the project with a greater understanding of how to write algorithms that will produce realistic collisions
- Finish the project within the deadline

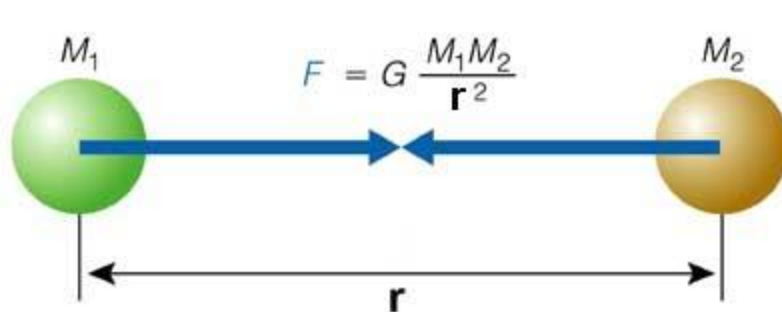
How are the positions of n particles calculated?

- ⦿ The position of particles is normally calculated using Newtonian dynamics
- ⦿ Therefore the positions of the particles can be calculated to be the superposition of the forces on the particles
- ⦿ The sum of the force on one particle with respect to all the other particles will give the net force on the particle and therefore it's new position and velocity

Equations Needed to Calculate the Motion of N Particles



$$\vec{F} = m\vec{a}$$



Calculations

- The coordinate system in the N-body simulation will be using the classic Cartesian coordinate system
- We will use a discretized time step in all of the calculations
- Therefore the force will be calculated using the equation $F_g = (GMm)/r^2$ or $F_e = (KQq)/r^2$ then broken into its components $F_x = ma_x$ and $F_y = ma_y$ and $F_z = ma_z$

Calculations

- ⦿ The method of finite difference approximations that will be used is called the velocity-verlet method
- ⦿ The first step in this method is to calculate the sum of the forces on the particle and break this up into the components f_x f_y and f_z
- ⦿ This will give the acceleration in the x,y,z directions respectively through Newton's second law $F_{\text{net}}=ma$

Calculations

- ⦿ The position of a particle in the x direction will be given by
- ⦿ $\text{Pos}_{x1} = \text{pos}_{xi} + v_{xi} * dt + 0.5a_{x1} * dt^2$
- ⦿ $\text{Pos}_{x2} = \text{pos}_{x1} + v_{x1} * dt + 0.5a_{x2} * dt^2$
- ⦿ $v_{x1} = v_{xi} + 0.5a_{x1} * dt^2$
- ⦿ pos_y and pos_z will be given by the same set of equations

Testing Algorithms

- Once a correct set of algorithms have been setup I will change the values for the initial position and velocity of N particles in order to verify that the algorithm is working correctly
- If time allows I will also try to run the experiment as if the particles that were interacting also had an initial charge(so two components gravity and electricity)

Timeline of Project

Date	Activity
10/21/14-11/6/14	Start to write code in Matlab, get correct equations of position, velocity and acceleration at time dt
11/7/14-11/20/14	Test code work through any logical errors, try manipulating initial conditions to verify code works properly
11/21/14-12/01/14	Begin report of project, analyze data and conclude findings
12/02/14	Submit report

References

- ⊙ <http://bh0.phas.ubc.ca/210/Doc/term-projects/kdv.pdf>
- ⊙ <http://laplace.physics.ubc.ca/210/TermProposals.htm>
- ⊙ <http://laplace.physics.ubc.ca/210/Proposals-2012/L1A.pdf>
- ⊙ http://en.wikipedia.org/wiki/N-body_simulation
- ⊙ <http://introcs.cs.princeton.edu/java/assignments/nbody.html>
- ⊙ http://en.wikipedia.org/wiki/Finite_difference_method
- ⊙ <http://webphysics.davidson.edu/Projects/SuFischer/node47.html>
- ⊙ <http://physics.princeton.edu/~fpretori/Nbody/intro.htm>

The Simulation of Simple Neural Networks

Jack Edwards

- **Overview**

- Neural network large array of interconnected units (neurons)
- Individually, each unit is very simple; connection to other units also very simple, positive or negative
- Large number N of neurons, N^2 connections, complex behaviour occurs
- Modeled with associated “spin” or energy (on/off), leading to lowest energy conditions where a “memory” is; Ising Model

- **Goals**

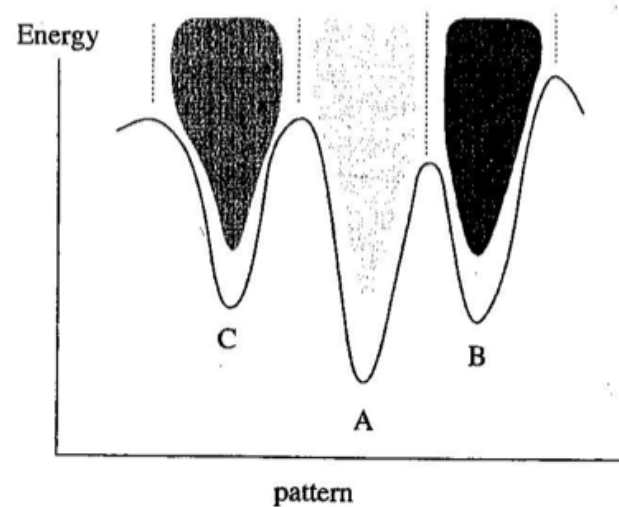
- Simulate neural network through MATLAB code
- Able to “see” and distinguish several patterns; memory
- Able to “learn” new patterns, “forget” old ones; creation of new memory

• Mathematical Formulation

– x x x
x x x
x x x

– Total energy of the network:

$$E = - \sum_{i,j} J_{i,j} S_i S_j \quad (1)$$



- Forms an energy distribution with “peaks” and “valleys”
- J is influence of neuron i on firing rate of j ; input J is negative, favourable for s_i to also be negative; J positive, s_i positive favoured
- Flipping of value s_i and s_j determined individually; if flip lowers energy, it occurs; if flip raises or keeps energy constant, it does not occur
- Desired pattern is stored as a minimum in the above equation; given an initial configuration, tends towards the nearest minimum; “recognition”

- Distance between each memory, Hamming distance:

$$\Delta_{m,n} = \frac{1}{N} \sum_i [s_i(m) - s_i(n)]^2 \quad (2)$$

- m, n particular patterns; $s(m)$ $s(n)$ is configuration of neurons in particular pattern, N is total neurons
- Larger value, greater chance of arriving at desired pattern; small distance, network can become “confused”
- Stable storage of pattern ‘ m ’

$$J_{i,j} = s_i(m)s_j(m) \quad (3)$$

- Insert into original equation:

$$E(m) = - \sum_{i,j} s_j(m)s_i(m)s_i s_j \quad (4)$$

- If already in configuration m , $s_j=s_j(m)$, $s_i=s_i(m)$, energy becomes large and negative; minimum reached

- Storage of multiple patterns:

$$J_{i,j} = \frac{1}{M} \sum_m s_i(m)s_j(m) \quad (4)$$

- m are stored patterns, M is total number of stored patterns; stored patterns have much lower energy than a random pattern
- Maximum practical stored patterns is $\sim N$; beyond, energy landscape becomes very indistinct; no clear “valleys”
- How can network “learn”?

$$J_{i,j}(new) = \beta J_{i,j}(old) + \alpha s_i(p)s_j(p) \quad (5)$$

- p is the new pattern to be stored; α is parameter that controls how fast new pattern is stored, β can be adjusted to allow for fading of old memories
- Connections $s_{i,j}$ change their strength to incorporate new pattern, forget old one

- **Approach**

- Simplest way to store the spin values of the neurons in a 2-D array, $\text{spin}(m,n)$; m, n correspond to row and column where particular neuron is located within network; size $N \times N$
- $J_{i,j}$, strength of interaction between neurons i and j , can be stored in a separate array; calculation from (3) done until all interaction energies have been calculated; array dimensions of $N^2 \times N^2$; discretized functions for $s_i(m)$ and $s_j(m)$ to simplify calculation
- E function of (1) determined using the calculated $J_{i,j}$ values

- **Testing and Experiment**

- Supply network with preselected memories; given pattern that is selected to be close to certain memory, returns the correct pattern
- Previous test conducted, but with many of the neural connections severed; determine maximum number of severed connections that still retains functioning network
- Train network to recognize new patterns and incorporate them into memory

- Timeline (subject to change)

Dates	Activities
Oct. 20-28	Research and basic code design
Oct. 29- Nov. 13	Implementation of code
Nov. 13-18	Testing of code
Nov. 19-23	Experiment and data analysis, begin report
Nov. 27	Finish report
Nov. 27	Hand in report (Deadline Dec. 3)

- References

N. Giordano and H. Nakanishi, *Computational Physics*, Addison-Wesley, Boston, (2005)

Crank Nicholson Approximation of the Schrödinger Equation

Phys 210 Term Project Proposal

Sanjaya Gebrial

Overview

- The 1-dimensional Schrödinger equation is a linear wave equation in 1 space variable and time.
- It can be solved to find the wave function of a particle which, after manipulation, gives the probability density to find a particle at any point in space and time.
- The Schrödinger equation can be expanded to 3 spatial dimensions.

Project Goals

- Solve the Schrödinger equation using the Crank Nicholson approximation, a second order implicit finite difference approximation.
- Error checking by comparison of simulation to known exact solutions and conservation of probability.
- Study behaviour of wave function by varying the initial conditions and the potential energy function.

Mathematical Formulation

- The 1-dimensional Schrödinger Equation:

$$i\hbar \frac{\partial}{\partial t} \Psi(x,t) = \frac{-\hbar^2}{2m} \frac{\partial^2}{\partial x^2} \Psi(x,t) + V(x,t) \Psi(x,t)$$

- $\Psi(x,t)$ will be solved as an initial boundary value problem with initial conditions

$$\Psi(x,0) = \Psi^0(x)$$

where $\Psi^0(x)$ is some specified function and $\Psi(x,t)$ has boundary conditions

$$\Psi(x_{\min}, t^n) = 0 = \Psi(x_{\max}, t^n), \quad n = 1, 2, \dots, n_t$$

- $V(x, t)$ will be some (real-valued) potential energy function.

Numerical Approach

- The Schrödinger equation will be discretized using a second order finite difference approximation where the continuum (x, t) will be replaced with discrete points (x_j, t^n) where

$$x_j = x_{\min} + (j - 1) \Delta x, \quad j = 1, 2, \dots, n_x$$

$$t^n = (n-1) \Delta t, \quad n = 1, 2, \dots, n_t$$

- I then approximate the Schrödinger equation to the form

$$i\hbar \frac{\Psi_j^{n+1} - \Psi_j^n}{\Delta t} = \frac{-\hbar^2}{2m} \frac{1}{2} \left(\frac{\Psi_{j+1}^{n+1} - 2\Psi_j^{n+1} + \Psi_{j-1}^{n+1}}{(\Delta x)^2} + \frac{\Psi_{j+1}^n - 2\Psi_j^n + \Psi_{j-1}^n}{(\Delta x)^2} \right) + \left(\frac{1}{2} (V_j^{n+1} + V_j^n) \right) \left(\frac{1}{2} (\Psi_j^{n+1} + \Psi_j^n) \right)$$

Numerical Approach(Continued)

- When given the potential energy function, $V(x, t)$, and the initial conditions for the wave function, $\Psi(x, t^1)$, the approximation becomes a set of equations in the unknowns $\Psi(x_j, t^{n+1})$ for $j=1,2,..n_x$.
- To solve for $\Psi(x_j, t^{n+1})$, we put the approximation equation into the form $A\Psi^{n+1} = B\Psi^n$.
- Solving for $\Psi^{n+1} = A^{-1}B\Psi^n$ gives us the next time step, at which point we repeat the process

Visualization and Plotting

- I will use Matlab's plotting facilities for plotting wavefunctions to be included in the report.
- I will also make animations, possibly with any matlab functions that are capable.

Testing and Numerical Experiments

- Ensure that $O(\Delta x^2)$ convergence behaviour is obtained by comparing results of discretization scales Δx , $\Delta x/2$, $\Delta x/4$, ... against known exact solutions for Ψ .
- Check that conservation of total probability is obeyed at all times t^n .
- Investigate behaviour of free particle in a box, 1-d tunneling, and 1-d resonance.
- Investigate differences between localized and non-localized initial wavefunctions.

Project Timeline

Dates	Activities
10/18 – 10/24	Basic Research, derive equations, begin code design
10/25 – 11/07	Implement code
11/08 – 11/12	Test Code, debug
11/13 – 11/17	Run numerical experiments
11/18 – 11/24	Analyze data
11/25 – 12/01	Write report, submit report

- **References**

- <http://www.dynamicearth.de/compgeo/Tutorial/Day2/cranknicholson.pdf>
- <http://laplace.physics.ubc.ca/210/Doc/term/schrodinger.pdf>

Comments?
Questions?
Suggestions?

Gravitational N-Body Simulation

Phys 210 Term Project Proposal

Robert Hill

Overview:

- Solving the positions of an arbitrary quantity(n) body of particles interacting gravitationally.
- Gravitational interactions will create an acceleration that is dependent on the location of the strongest acting force on the particle. Things such as energy, momentum, and angular momentum are all things that need to be conserved. Things such as friction, damping, or any other fundamental forces are ignored for this simulation.

Project Goals:

- To create a simulation that properly visualizes how a large body of particles would interact with one another on a gravitational level.
- Once simulation is complete, test to make sure the code runs as theory states it should.
- Test the simulation through a variety of scenarios, and that in all cases the simulation runs as it should.

Mathematical Formulation

- .At any point, the force on a particle due to gravity is defined as:

$$a(t) = \sum_{i=1}^n GM_i / r_i \quad \text{and} \quad x(t) = \iint a(t) d^2t$$

Numerical Approach:

- I am currently in the process of reading on how n-body simulations should work. Obviously, there will be calculations of the gravitational force on individual particles, but the methods of numerical approximation are unknown to me as of yet.

Visualization and Plotting tools:

- Besides any default methods of visualization that are available in MATLAB and OCTAVE, I am not sure which methods I will be using.

Testing and Numerical Experiments:

- Energy and momentum conservation. Since I do not intend to include any forms of friction, and most likely any other forms of damping, the sum of all energies and momenta should be constant. Any violation of this would point out a flaw in my code.
- Visual testing. This is the most obvious method of testing. If a particle starts acting in a way it shouldn't, that would also point out a code flaw.

Project Timeline:

Dates	Activities
Oct 24-Oct 31	Initial research and derivations
Nov 1 –Nov 15	Code construction
Nov 16 - Nov 20	Test Code
Nov 21 – Nov 25	Run code through multiple Experiments; Analyze data; Begin Report
Nov 25 - 30	Finish Report
Dec 1	Submit Report;

References:

- Greenspan, D. (2004). *N-body problems and models*. Singapore: World Scientific.

Chaos in Chua's Circuit

Michael Horner

Oct. 21, 2014

Overview

Chaos implies sensitivity to initial conditions - “butterfly effect”

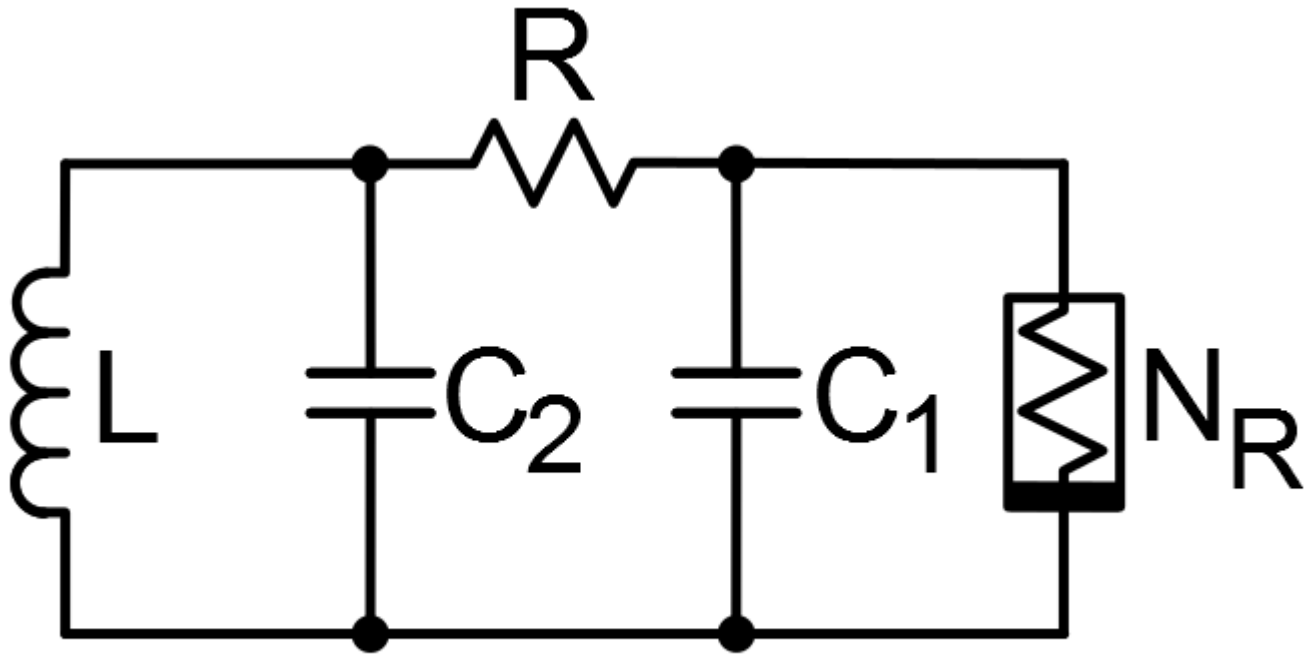
Chaotic oscillators display non periodic-behaviour, with some “attractors”

Examples – Weather system -> Storms

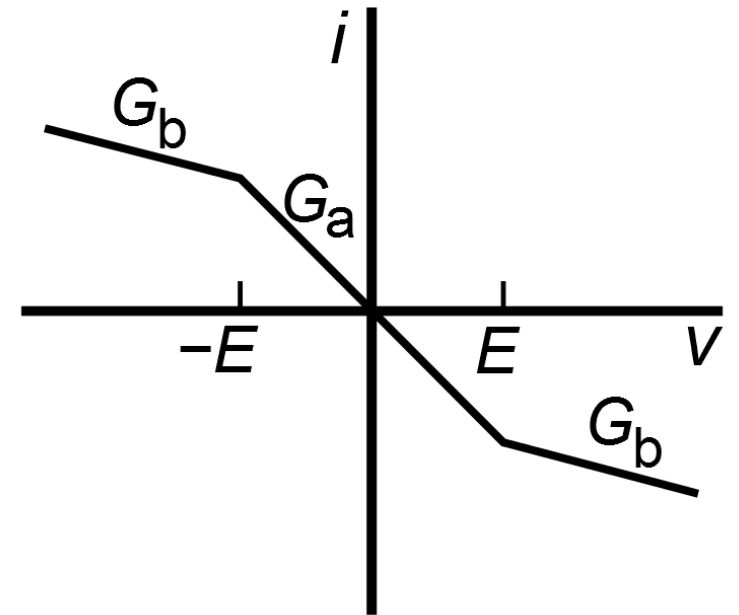
- Lorenz System -> Lorenz Attractor

- Chua’s circuit -> “Double scroll” Attractor

Simplified Circuit Diagram

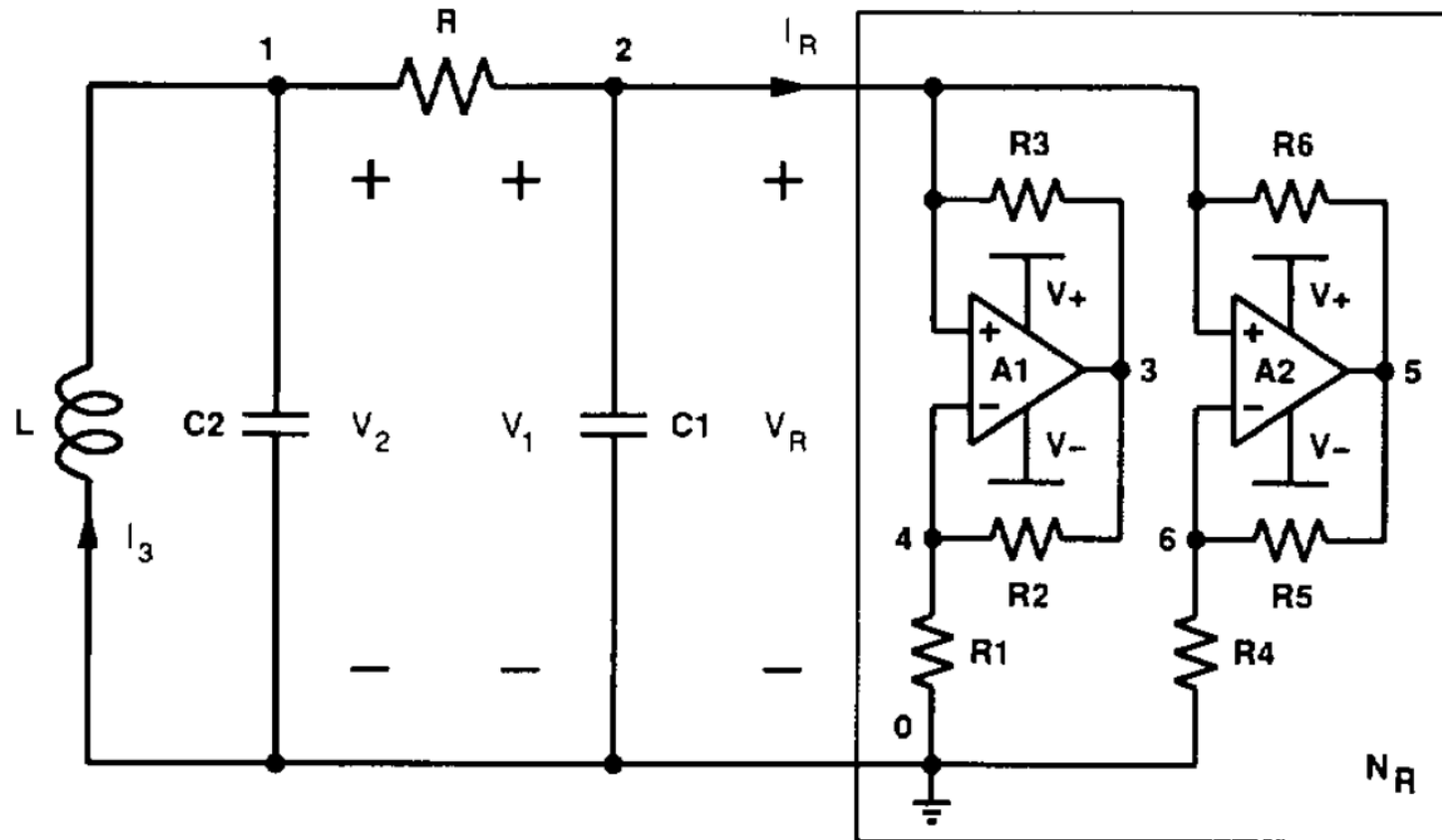


- 1 x Inductor
- 2 x Capacitor
- 1 x Nonlinear Resistor



Required nonlinear resistor behaviour

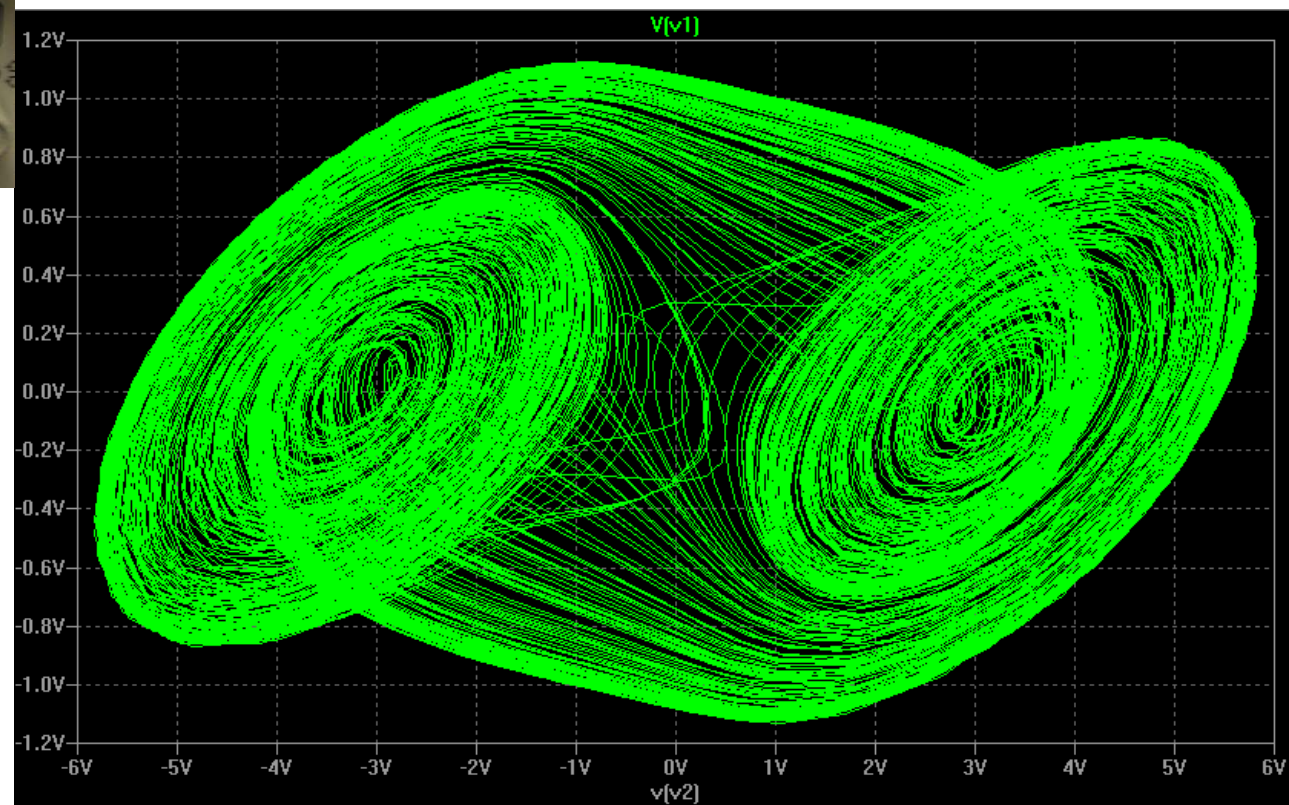
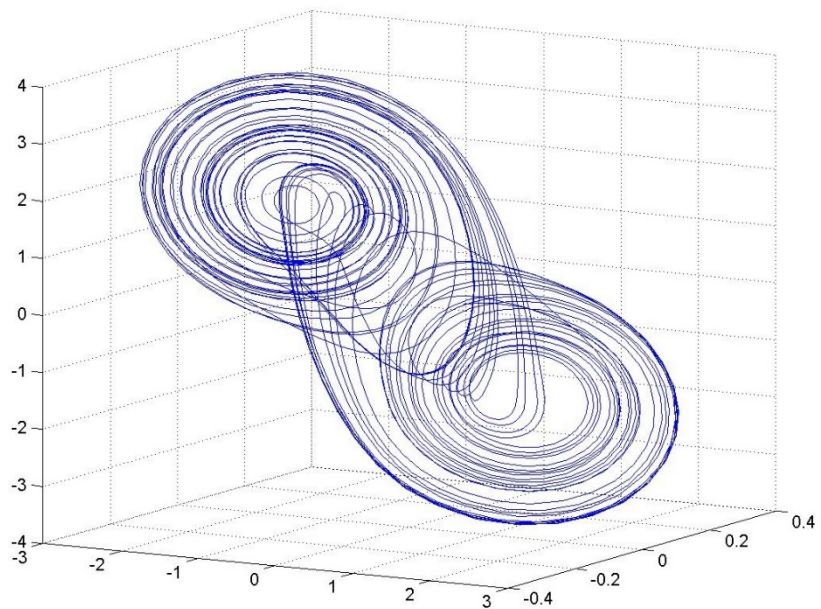
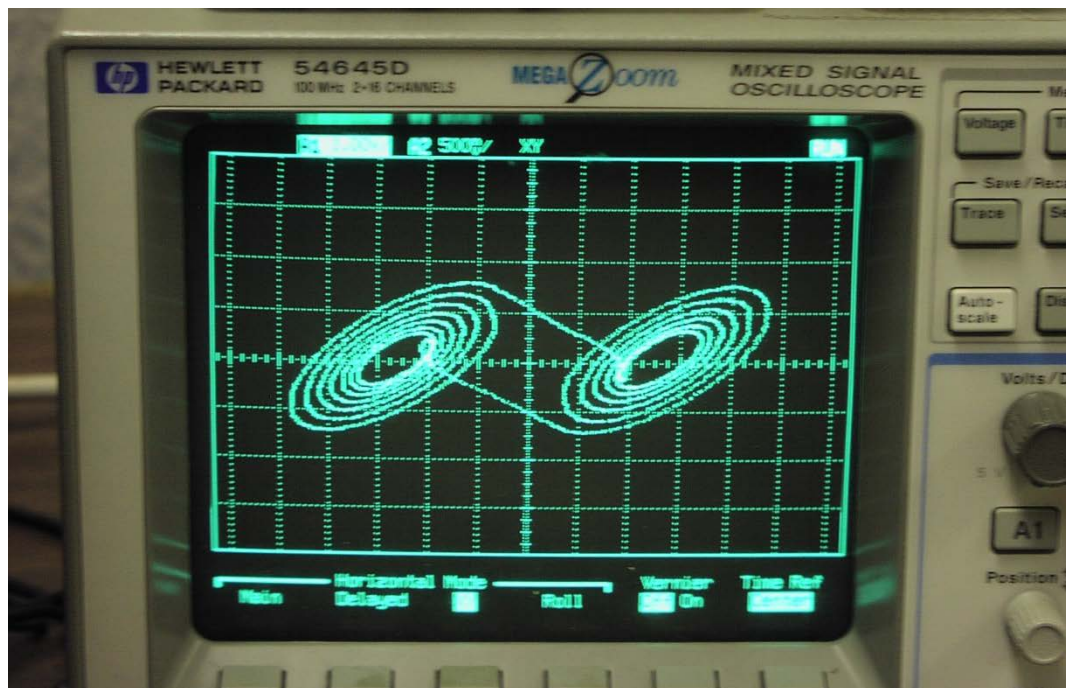
Real Implementation



Practical implementation of Chua's circuit
using two op-amps

Project Goals

- 1) Implement the Chua's circuit model in MATLAB
- 2) Generate plots that display the chaotic behaviour
- 3) Investigate a variety of parameters for components
- 4) Look for patterns, possibly find mathematical relationships



Mathematical Formulation

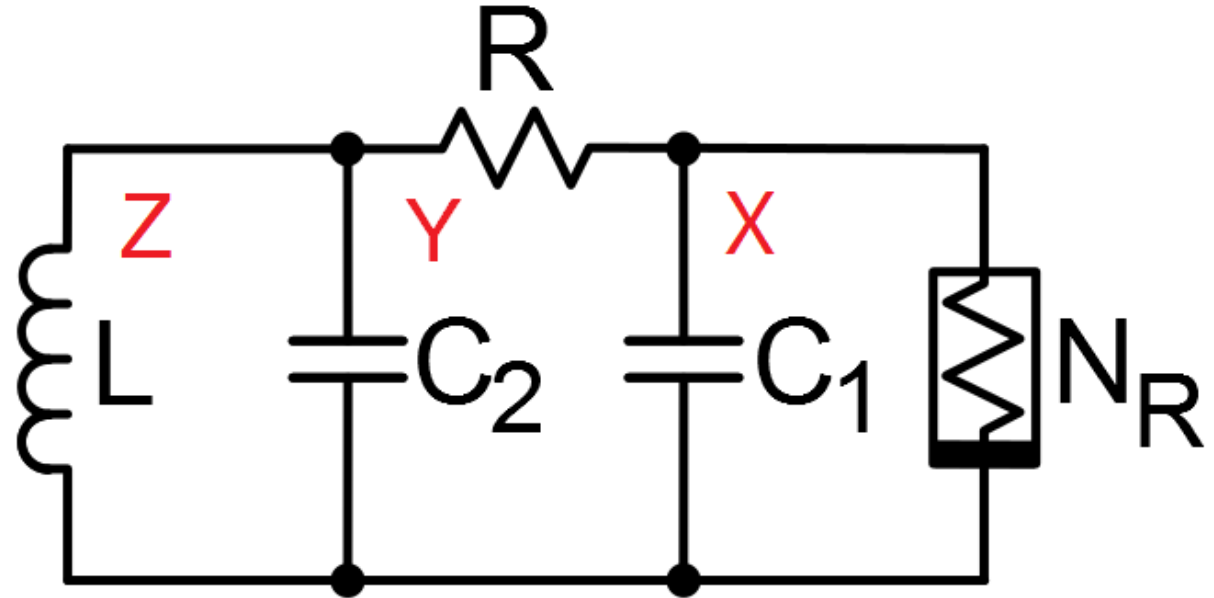
$$\frac{dx}{dt} = \alpha[y - x - f(x)]$$

$$\frac{dy}{dt} = x - y + z$$

$$\frac{dz}{dt} = -\beta y$$

$f(x)$ = nonlinear resistor response

x, y, z all functions of time, α, β
component parameters (L or C)



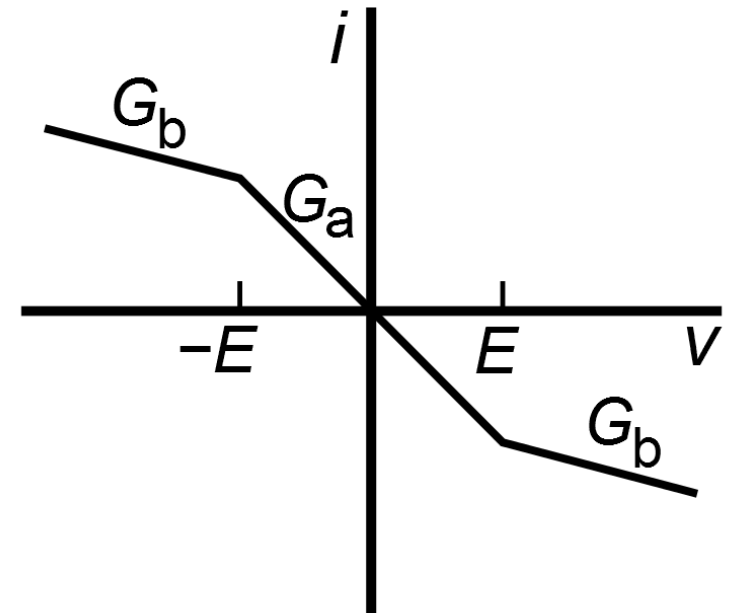
Units = volts

Mathematical Formulation (cont'd)

$$f(x) = m_0 + \frac{1}{2}(m_1 - m_0)(|x + 1| - |x - 1|)$$

Where $m_0, m_1 < 0$

Note: Diagram shows case with units, simulation will be dimensionless for simplicity.



Numerical Approach

- MATLAB ODE solver (ode45)
- Runge - Kutta method
- 4th order method

Visualization and Plotting Tools

- Matlab's plotting facilities (2D/3D) for basic plots
- Possible video showing time evolution of chaotic trajectory

Testing and Experiments

-Varying of m_0, m_1, α, β

-Changing behaviour of nonlinear resistor ($f(x)$)

Project Timeline

Preliminary Research – Which method to use? What to look for?	Oct. 20 th – Oct. 24 th
Code implementation + testing	Oct.25 th – Oct. 31 st
Begin analysis and data collection	Nov. 1 st – Nov.14 th
Start report	Nov. 15 th
Finish Analysis	Nov.16 th – 21 st
Finish report	Nov. 22 nd –Dec.2 nd
Hand in Project	December 3 rd

References

Jackson, L. B., A. G. Lindgren, and Y. Kim. "A chaotic attractor from Chua's circuit." *IEEE Transactions on Circuits and Systems* 31.12 (1984).

Kennedy, Michael Peter. "Three steps to chaos. I. Evolution." *Circuits and Systems I: Fundamental Theory and Applications, IEEE Transactions on* 40.10 (1993): 640-656.

<http://www.electronics-lab.com/blog>

<http://www.mathworks.com/help/matlab/ref/ode45.html>

<http://nonlinear.eecs.berkeley.edu/>

Comments or questions

Gravitational N body problem, including toomre model

Phycis 210 term project proposal
By Jiawei Huang

Overview:

Gravitational N-body simulations, that is numerical solutions of the equations of motions for N particles interacting gravitationally.

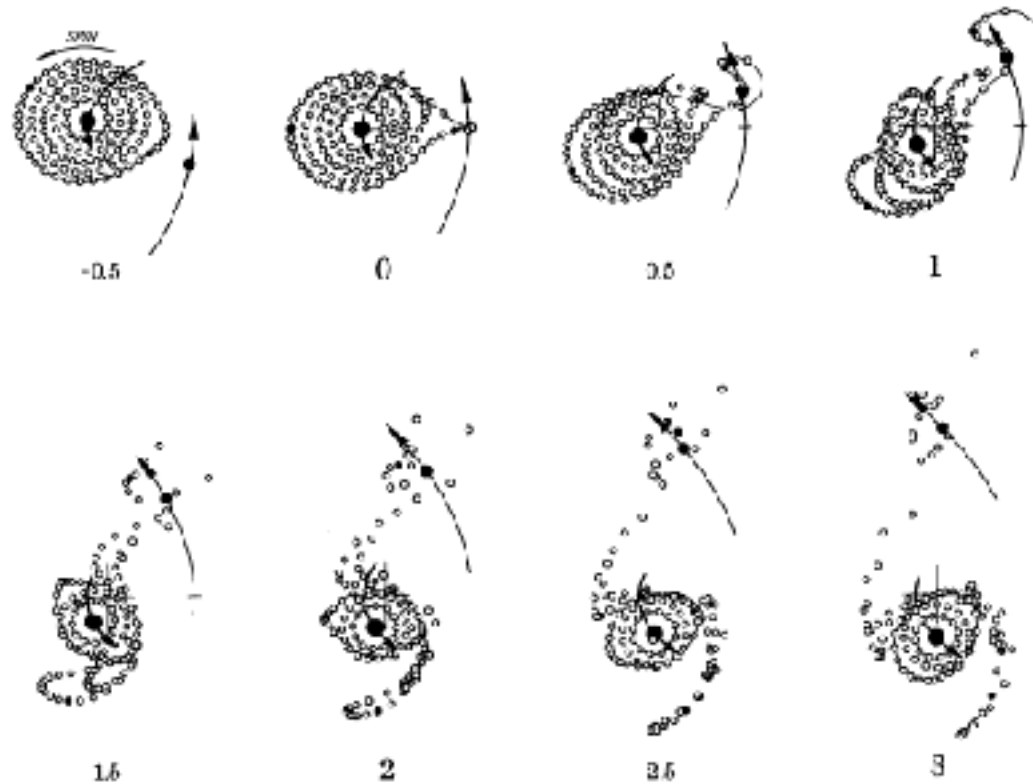
This method are widely used in astrophysics, with applications from few body or solar system like systems all the way up to galactic and cosmological scales.

Project goals:

Use a matlab (spyder) code to depict the collision of n bodies.

Visualize the n-body motion with MPEG files.

The simulation by Toomres of a low mass companion passing close to a disk galaxy



Mathematical Formulations

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

Gravitational Force

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

Centripetal force

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

Kepler's 3rd law:

P - period of the planet orbit

G - gravitational constant (6.67 x 10⁻¹¹ Newton m²/kg²)

M₁ - mass of particle 1

M₂ - mass of particle 2

a - distance between two particles

d - length of the semi major axis of the ellipse orbit

$$P^2 = d^3$$

Numerical Approach

Use FDA – finite difference approximation
to calculate the position and velocity for each particles

$$f(x) = \sum_{k=0}^{\infty} \frac{\Delta^k[f](a)}{k!} (x - a)_k = \sum_{k=0}^{\infty} \binom{x - a}{k} \Delta^k[f](a) ,$$

- * Consider 3 dimensional vectors for all variables

Visualization and plotting

- * Using matlab (spyder) to output the plotting graphs and MPEG.

Testing and numerical experiment

- * All particles will be given different mass (randomly)
- * The initial velocity of all particles will be given 0
- * All particles will have different initial position
- * Gradually increasing n to as large as it can be
- * Attempt to depict the toomre model

Project timeline

Dates	Activities
Oct 19-Oct 25	Research and design code
Oct 25-Nov 13	Implement code
Nov 13-Nov 22	Test code
Nov 22-Nov 28	Running experiment and starting report
Nov 28-Dec 1	Finish report
Dec 2	Submit report

Reference

- * Dr. Beverly Smith, East Tennessee State University
Department of Physics, Galaxy Collisions,
<http://faculty.etsu.edu/smithbj/collisions/collisions.html>
- * [http://www.scholarpedia.org/article/N-body_simulations_\(gravitational\)#Introduction](http://www.scholarpedia.org/article/N-body_simulations_(gravitational)#Introduction)
- * <http://www.rsmas.miami.edu/personal/miskandarani/Courses/MSC321/lectfiniteDifference.pdf>

Questions

Ray Tracing and Optics Simulations

Term Project Proposal

Chironjeev Kanjilal

PHYS 210

October 21st 2014

Overview

- Light can reflect off of surfaces or refract through mediums
- The velocity and path of light depend upon the medium through which it travels
- Path of light can be traced easily with objects like lenses, prisms, and mirrors
- Can also be applied to more complex systems with irregular or changing mediums
- Applications include radio signal tracing and propagation, ocean acoustics, seismology, and optical design

Project Goals

- Study an optical system and the different ways in which light interacts with each component
- Create and implement a MATLAB code which traces the path of multiple rays of light
- Create an adaptable system of lenses, mirrors, and prisms
- Investigate a variety of different systems and their applications

Mathematical Formulation


- Snell's Law
$$\frac{\sin \theta_1}{\sin \theta_2} = \frac{v_1}{v_2} = \frac{n_2}{n_1}$$
 - Total Internal Reflection, Critical Angles
- Fresnel Equations
- Lensmaker's Equation
$$P = \frac{1}{f} = (n - 1) \left(\frac{1}{R_1} - \frac{1}{R_2} + \frac{(n-1)d}{nR_1R_2} \right)$$
- Ray Transfer Matrices
- Law of Reflection
$$\theta_i = \theta_r$$
- Dot and Cross Product
$$A \cdot B = ||A|| ||B|| \cos \theta$$
$$||A \times B|| = ||A|| ||B|| \sin \theta$$
- Classical Kinematics
- Multiple Reflections, Retroreflection, Diffuse Reflection

Project Timeline

Dates	Task
October 22 nd – November 31 st	<ul style="list-style-type: none">• Code research• FDA derivations• Experiment with equations and code• Decide on optical systems to consider
November 1 th – November 15 th	<ul style="list-style-type: none">• Begin implementation of code
November 16 th – November 25 th	<ul style="list-style-type: none">• Test code• Run numerical experiments• Test optical systems• Analyze data• Begin writing report
November 26 th – November 30 th	<ul style="list-style-type: none">• Finalize visualizations• Study conclusions• Continue report writing
December 1 st	<ul style="list-style-type: none">• Project submission

Questions or Comments?

Thanks For Your Time



Optics: 2-D Ray tracing through a series of lenses, prisms, and mirrors

Phys 210 Term Project Proposal

Pawel Kapusta

Overview

- Light can be represented as a ray which travels through different mediums and interacts with various objects around it
- When light interacts with simple objects such as mirrors, prisms, and lenses these interactions can be represented with mathematical formulas

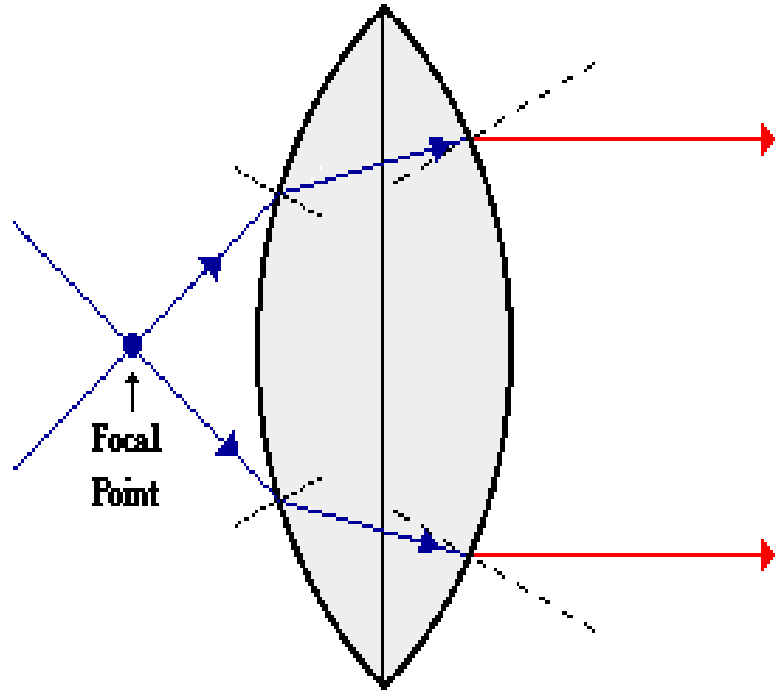
Goals

- To have a visual representation of these interactions with light and some simple objects
- To investigate a range of initial conditions that can be varied and generate a hypothetical outcome through the use of MATLAB (Octave)

Mathematical Formulation

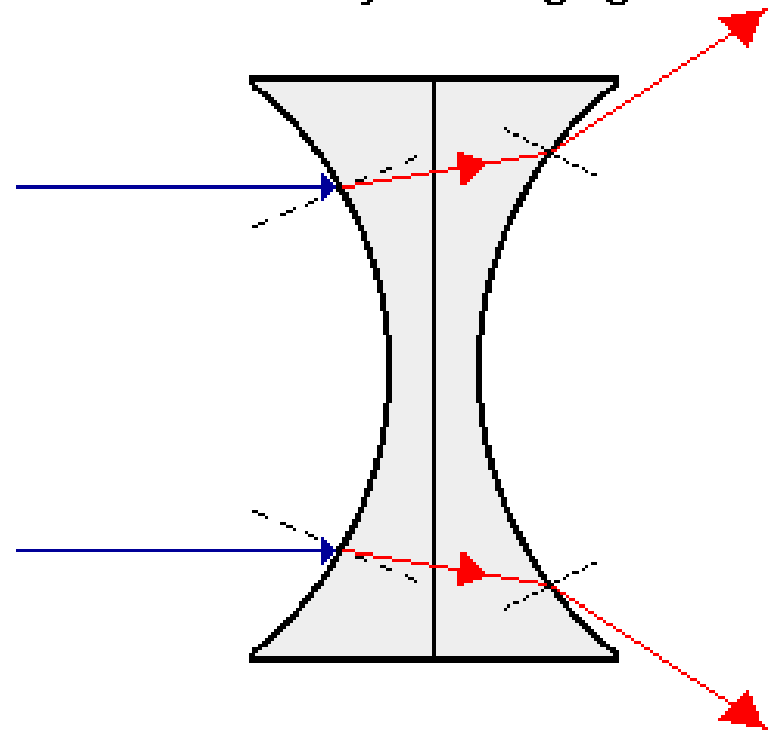
- For any mirror use the Law of Reflection: $\theta_i = \theta_r$
- The angle the ray hits at is the angle at which it reflects at
- For refraction use Snell's Law: $\sin(\theta_1)n_1 = \sin(\theta_2)n_2$
- n_1 will be the index of refraction in air and n_2 will be the index of refraction in crown glass
- $n_1 = 1.000293$, $n_2 = 1.56$
- Can find out how a light ray interacts with any mirror, prism, or lens by applying Snell's law and the law of reflection

Refraction by a Converging Lens

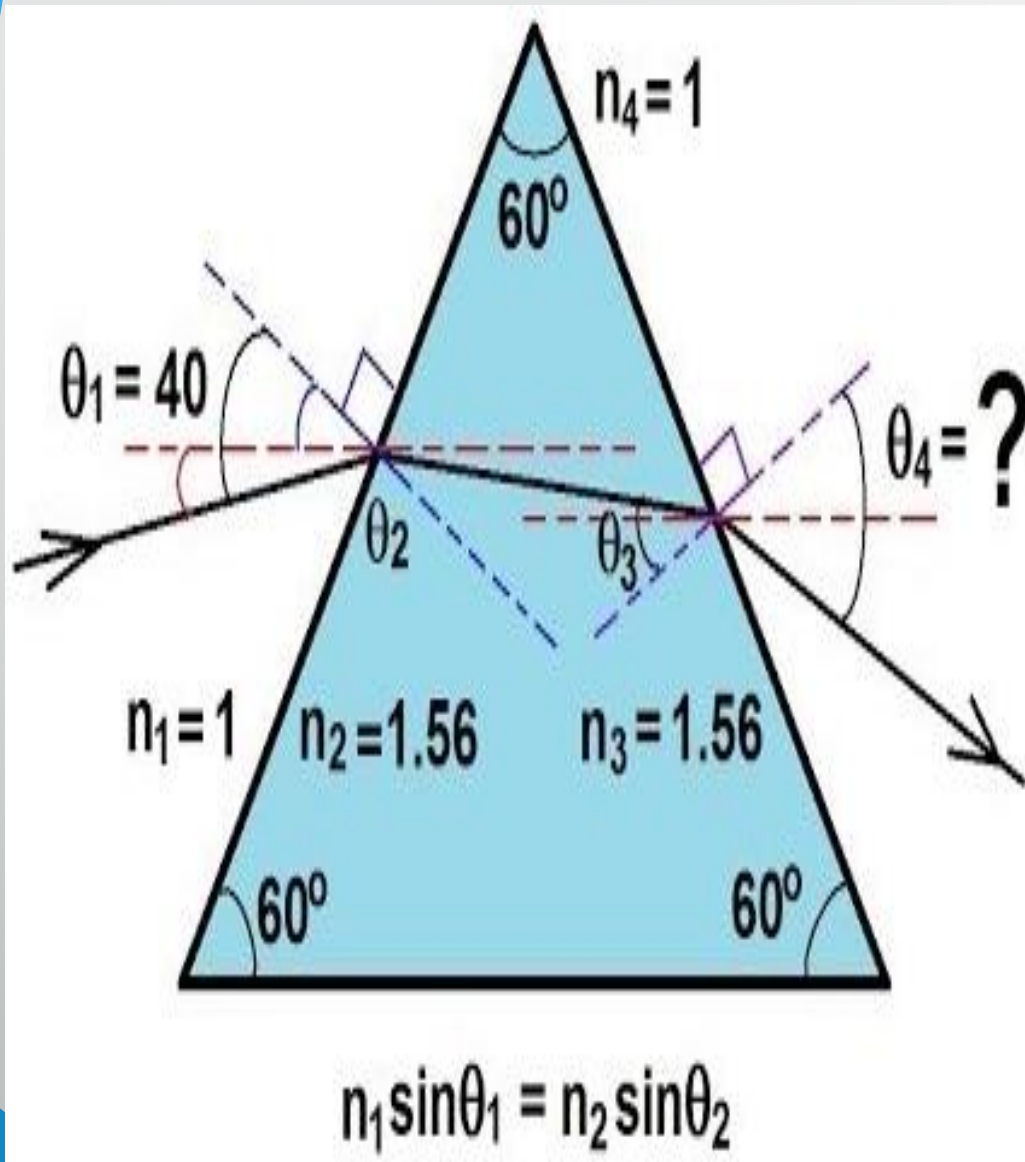


Incident rays which travel through the focal point will refract through the lens and travel parallel to the principal axis.

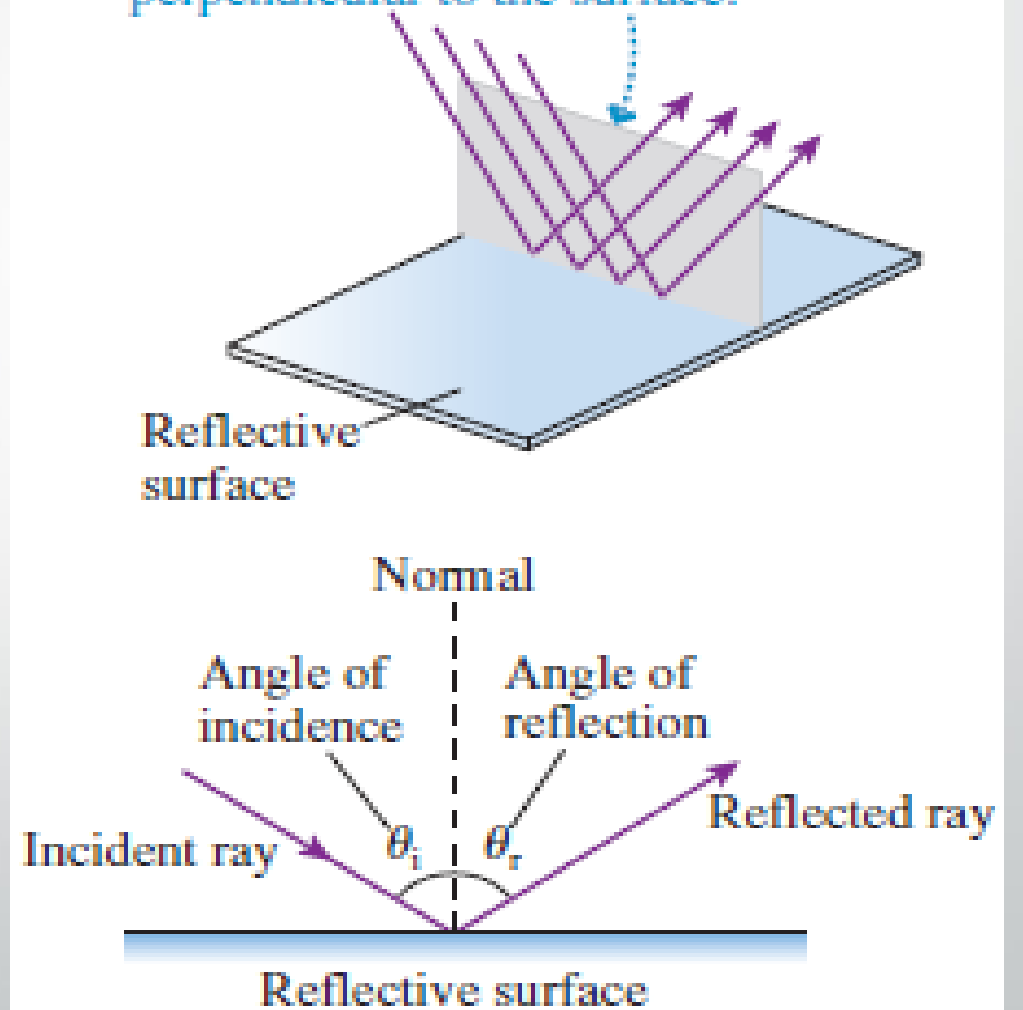
Refraction by a Diverging Lens



Incident rays traveling parallel to the principal axis will refract through the lens and diverge, never intersecting.



The incident and reflected rays lie in the plane of incidence, a plane perpendicular to the surface.

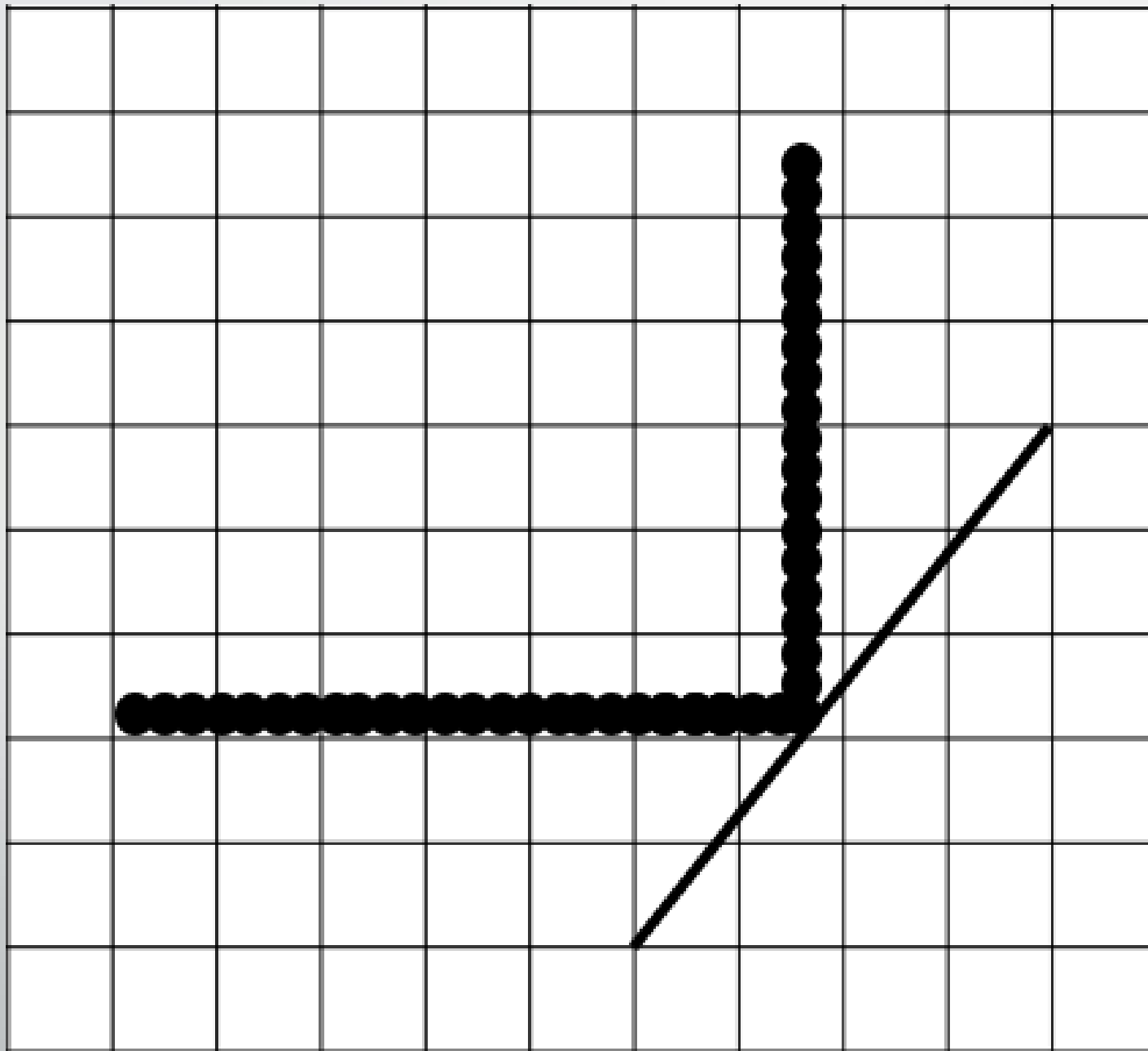


Numerical Approach/Logic

- Initially create a 2 dimensional Cartesian plane
- Then place mirrors, prisms, etc on the plane with known coordinates
- Ex. Can place a mirror by using a line segment with the equation $y = 2x$ on a domain from 5 to 7
- Can then model a ray of light by placing a particle on the coordinate system and giving it some direction vector
- Ex. Particle is placed at $(x,y) = (1,2)$ and given a direction vector of $\langle 3, 1 \rangle$
- Then as simulation is running, increment x by some finite value of time and visually record where the particle was for each increment of x .

Numerical Approach/Logic Continued

- As the simulation is running, have a check for each increment of x as to whether the particles coordinates coincide with any of the mirror's, prism's, or lens' coordinates.
- If they do, then calculate the normal of the object and the angle at which the particle's direction of movement is from the normal
- Then apply Snell's law $\sin(\theta_1)n_1 = \sin(\theta_2)n_2$
- $\theta_1, n_1,$ and n_2 are all known, thus can solve for θ_2
- Find particles new direction vector
- Rinse and repeat for a set time frame!
- NOTE: The walls of the coordinate system will be treated as mirrors so that the particle is trapped



Project Timeline

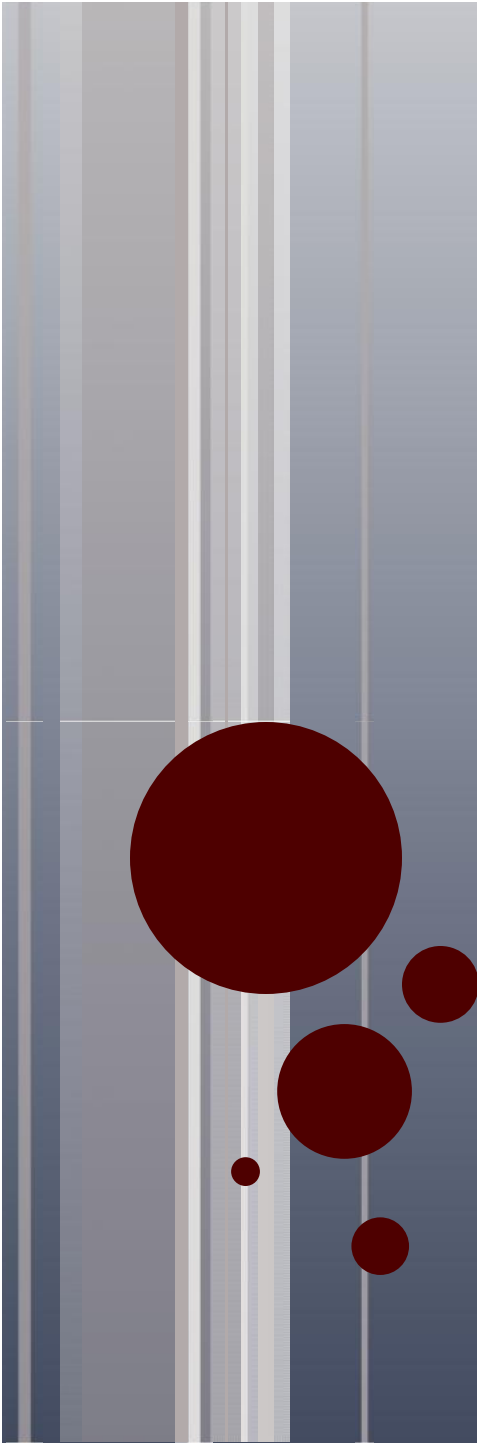
Dates	To Do
Oct 15 – Oct 24	Do basic research and begin code design
Oct 25 – Nov 10	Implement Code
Nov 10 – Nov 15	Test Code
Nov 15 - Nov 21	Run numerical experiments, analyze data, begin report
Nov 21 - Nov 28	Finish Report
Nov 28 - Nov 30	Final Check
Dec 1	Submit Project

References

- <http://hyperphysics.phy-astr.gsu.edu/hbase/geoopt/refr.html>
- http://www.physics.fsu.edu/courses/summer06/phy2054c/Notes/Lect13-ch23_2.pdf
- <http://www.physicsclassroom.com/class/refln/Lesson-1/The-Law-of-Reflection>
- <http://www.physics.wustl.edu/ClassInfo/316/Theory/Refraction.pdf>
- <http://www.physchem.co.za/OB11-wav/lens2.htm>
- http://en.wikipedia.org/wiki/Snell's_law
- <http://www.physicsclassroom.com/class/refrn/Lesson-5/Refraction-by-Lenses>



Questions?
Comments?
Queries?



ELECTROSTATIC INTERACTIONS OF N CHARGES ON A SPHERE: THE THOMSON PROBLEM

Physics 210 Term Project Proposal

Paulette Kubik

OVERVIEW

- Charged particles placed on a sphere exert forces on each other by Coulomb's Law
- These charges will move about the sphere until static equilibrium is reached



PROJECT GOALS

- Write a Matlab code that uses FDA's that will demonstrate the charge distribution
- Create a simulation of the particle behaviour in 3D
- Examine how different variables affect the distribution
- Is there more than one equilibrium position?



MATHEMATICAL FORMULATION

- Coulomb's Law (vector form)

$$F = -\frac{q}{4f\epsilon_0} \sum_{i=1}^N q_i \frac{\mathbf{r} - \mathbf{r}_i}{|\mathbf{r} - \mathbf{r}_i|^3}$$

- Friction/Drag force (= drag, an adjustable variable)

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



NUMERICAL APPROACH

Simplifications

- Simplify charge values to be either +/-1
- The radius will be 1 (no effect on equilibrium)
- Charges will have equal mass
- Initial velocity will be set to Zero



VISUALIZATION AND PLOTTING TOOLS

- Will use Matlab for Visualization and Plotting

Testing and Numerical Experiments

- Can visually check equilibrium positions for situations with few particles ($N \leq 6$)
- Ensure charges stay on the sphere



PROJECT TIMELINE

Dates	Activities
Oct. 20 – Oct. 29	Research, understand use of equations, begin code design
Oct. 30 – Nov. 17	Create/implement code
Nov. 18 – Nov. 20	Test code
Nov. 21 – Nov. 25	Experiment, analyze data
Nov. 23 – Nov. 29	Write and submit report

References

<http://laplace.physics.ubc.ca/210/Doc/fd/nbody.pdf>

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





THE NEURAL NETWORK

A SIMULATION

PHYS 210 TERM PROJECT PROPOSAL

ESSENCE LAI

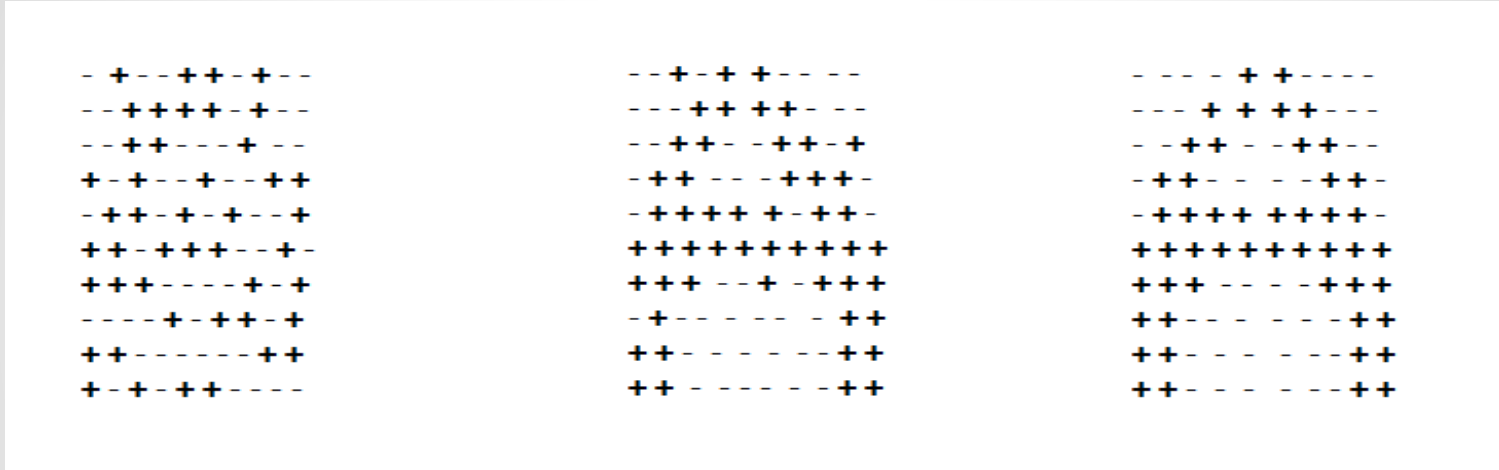
OVERVIEW

- Neurons are the basic unit of the neural network and have three parts: the body (called the soma), along with dendrites and an axon. They are connected by orientation of their given spin s_i and interacts with other spins by their interaction energy $J s_i s_j$
- The network operates as a “content addressable memory”. It recalls the developed pattern that is initialized by the system.

PROJECT GOALS

- To write a MATLAB(octave) code that simulates a simple neural network
- To simulate the neurons on/off behavior by the Ising spin model where each neuron has a spin of $s = \pm 1$
- To use the energy function to describe our neural network as a Ising model
- To use the Monte Carlo method to simulate the energy function through the task of display operation which should eventually display the desired states of each spin.

- To use a step function to show the gradual stabilization through the Monte Carlo pass.
- An example of our simulation is a 10 x 10 lattice of spins;



Initial pattern

After one Monte Carlo

After two Monte Carlo
sweeps

In decreasing probability

Mathematical Formulation

Interaction energy equation where m is the pattern:

$$J_{i,j} = s_i(m)s_j(m) \quad (1)$$

Effective energy of neural network:

$$E = - \sum_{i,j} J_{i,j} s_i s_j \quad (2)$$

Most energy stable pattern

$$E(m) = - \sum_{i,j} s_j(m) s_i(m) s_i s_j \quad (3)$$

For many patterns where M is the total number of patterns:

$$J_{i,j} = (1/M) \sum_m s_i(m) s_j(m) \quad (4)$$

Spin number from row and column numbers, m and n :

$$i = N(m-1) + n \quad (5)$$

Adding a new pattern where $s(p)$ is the new pattern:

$$J_{i,j}(\text{new}) = \beta J_{i,j}(\text{old}) + \alpha s_j(p) s_i(p) \quad (6)$$

TESTING

- I will construct a lattice and damage the percent of interaction energies to zero . This will play the base case as we interact with different values of the interaction energies and different number of neurons.

NUMERICAL ANALYSIS

- investigate interaction between having more neurons and the steps taken to meet the desired state.
- investigate the accuracy of determining the proper desired state from two very similar stored patterns

PROJECT TIMELINE

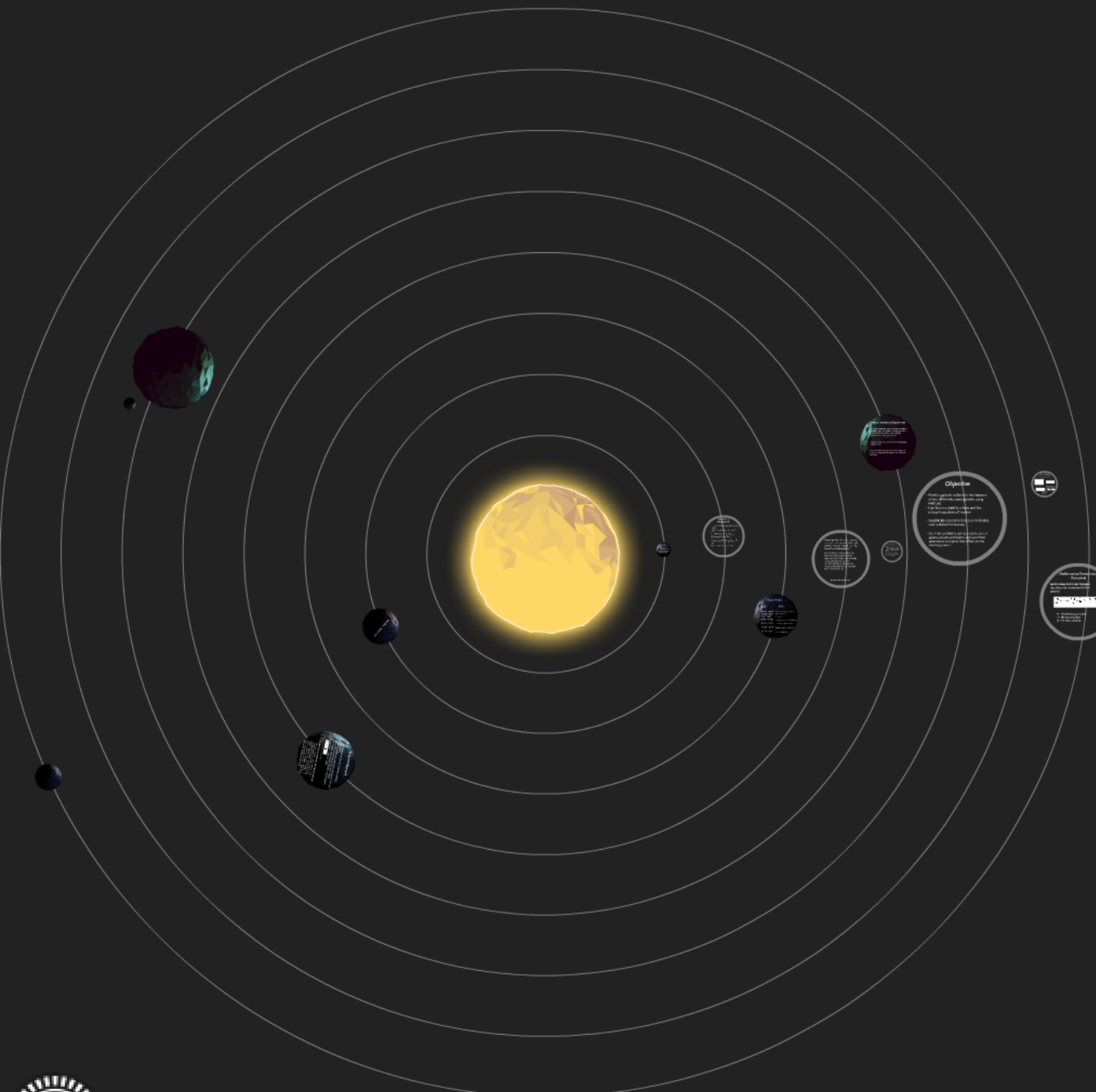
Dates	Activities
10/13 - 10/24	Do basic research and begin designing code
10/25 - 11/15	Implement code
11/ 16- 11/19	Test code and run numerical experiments
11/20 - 11/25	Analyze data and begin report
11/26 - 11/28	Finish report
11/28	Submit project

REFERENCE

-N.J. Giordano and H.Nakanishi, Computational physics, 2nd Edition, Prentice Hall, West Lafayette, 2005

-http://www.mathworks.com/help/pdf_doc/nnet/nnet_ug.pdf

THANK YOU FOR LISTENING!!!

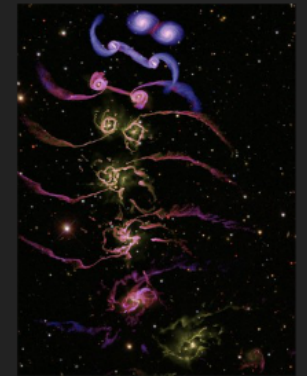


Toomre Model for Galaxy Collisions

Johnny
Lazazzera
10/20/2014

Physics 210 -
Introduction to
Computational
Physics

UBC



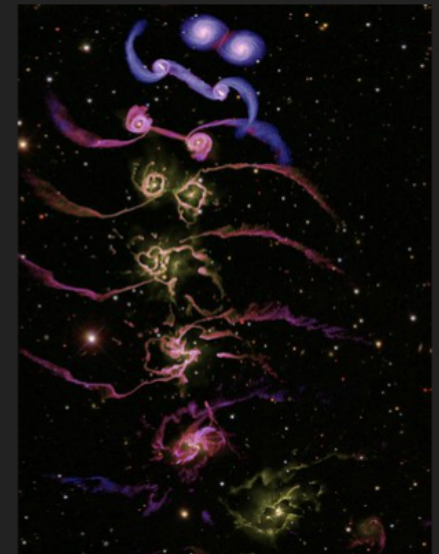
[MPI 2005] http://www.mpa-garching.mpg.de/mpa/research/current_research/hl2005-2b/hl2005-2b-en.html

Toomre Model for Galaxy Collisions

Johnny
Lazazzera
10/20/2014

Physics 210 -
Introduction to
Computational
Physics

UBC



[MPI 2005] http://www.mpa-garching.mpg.de/mpa/research/current_research/hl2005-2b/hl2005-2b-en.html

Overview / Background

- Question: What happens when galaxies collide?
- Start with two disc shaped galaxies very far apart
- For simplicity, these galaxies will be a collection of non-interacting particles in orbit around the center of mass.
- As the galaxies approach each other in orbit, tidal forces begin to cause spiral tails to form
- Results depend on size, speed and angle

- These 'particle' stars having mass less than the galaxy center will exhibit motion based on the forces from both galaxies.
- Just as tides on either end of the Earth experience gravitational force from the moon, the n-bodies of galaxies experience non-uniform tidal force wherein the nearest side experiences a greater force than the further.

But how do we know?

The Toomre Model

- 1964 - Alar Toomre proposes instability criterion
- Put simply: with information of the gaseous dispersion velocity and surface density one can predict the gravitational stability of galaxies.
- 1970 - The Toomre brothers, using computer simulation, demonstrate a galactic collision whereby the predicted tidal tail effect is displayed.

Objective

- Model a galactic collision in the instance of two differently sized galaxies using MATLAB.
- Use Toomre stability criteria and the relevant equations of motion
- Graphically represent forces on N-bodies over a distinct time scale
- Set initial conditions such as velocity, size of galaxy and axis orientation and vary these parameters to explore their effects on the resulting system

el
es instability
n of the gaseous
ace density one
stability of
s. using
onstrate a
he predicted



Mathematical Formulation (simple)

Newtons laws of gravitational force / motion

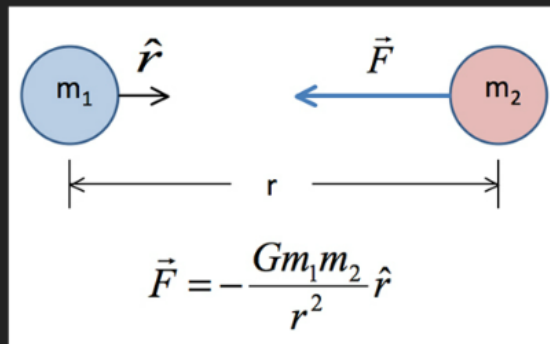


Image from: www.learner.org

Circular orbit:

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

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

Rewrite above eqn. get acceleration for every point mass:

$$\mathbf{a} = \mathbf{G} \times \mathbf{m}_1 / \mathbf{r}^2$$

Where M_1 (mass of galaxy) \gg M_2 (mass of star)
take $M_2 = 0$

Mathematical Formulation (Complex)

Collisionless Boltzman Equation:
describes star movement within
galaxies

$$\frac{d\mathbf{r}_i}{dt} = \mathbf{v}_i \text{ and } \frac{d\mathbf{v}_i}{dt} = \sum_{j \neq i}^N Gm_j \frac{\mathbf{r}_j - \mathbf{r}_i}{|\mathbf{r}_j - \mathbf{r}_i|^3}$$

N = # point masses (stars)
 \mathbf{r}_i = i th mass position
 \mathbf{v}_i = i th mass velocity

Numerical Approach

1. Set up initial 3D vectors (position, velocity, acceleration)
2. With coordinates and velocity vectors for each star at $f(x,v,t_0)$ calculate $f(x,v,t)$ at several times.

$f(x,v,t)dx dv$ = distribution function = mass at point (x,v) at time t = mass in $dx dv$

$$dv_i / dt = -\nabla \phi_i$$

Image from: <http://ned.ipac.caltech.edu/level5/Barnes/Barnes2.html>

3. Find dynamic equation for distribution function

This will involve using several more complex equations such as the equation for gravitational potential (Poisson equation), Euler's equation for time variation of velocity, and the continuity equation or mass conservation. Most of my time will be allocated to learning this approach.

Testing / Numerical Experiment

- Test several different values for initial conditions including angle/orientation, velocity, size and shape. Analyze the effect on small body interactions as well as galaxy cores.
- Try to maximize size of N without risking timely computations
- Use finite differencing for numerical approach (also allocating much of my time to understand this task)

Project Timeline

Date:

Goal:

10/21 - 10/27

Basic research, derivations, begin code

10/28 - 11/6

Implementing code

11/7 - 11/15

Testing code

11/15 - 11/20

Run the experiment, collect data, begin report

11/20 - 11/27

Analysis of data / Report / Presentation

11/27 - 11/29

Finalize presentation / complete draft

12 / 1 - 12/4

Present / Submit project

References:

<http://www.cv.nrao.edu/~jhibbard/students/CPower/dynamics/cbe/cbe.html>

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

<http://adsabs.harvard.edu/doi/10.1086/151823>

http://en.wikipedia.org/wiki/Galactic_tide#Galaxy_collisions

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

<http://publications.lib.chalmers.se/records/fulltext/134167.pdf>


http://www.mpa-garching.mpg.de/mpa/research/current_research/hl2005-2b/hl2005-2b-en.html

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

ned.ipac.caltech.edu/level5/Barnes/Barnes2.html



Questions / Comments?



Finite Difference Approximation of the Time-Dependent Schrödinger Equation

PHYS 210 Term Project Proposal

Samuel Leutheusser

Overview

- 🌐 The time-dependent Schrödinger Equation is a partial differential equation with a first-order time derivative and second order space derivative.
- 🌐 This equation is the quantum mechanics equivalent of Newton's 2nd Law in classical mechanics as it specifies a wavefunction that completely describes a particle state (or a collection of particles) and its time evolution

Project Goals

- 🌐 MATLAB code will be written to solve this differential equation numerically, using the Crank Nicholson finite difference approximation, for both time-dependent and time-independent potentials
- 🌐 The code will be tested using well known systems, such as the infinite square well
- 🌐 Various initial conditions will be used to characterize the behaviour of these wavefunctions

Mathematical Formulation

- 🌐 The 1-dimensional equation is:

$$i\hbar \frac{\partial}{\partial t} \Psi(x, t) = -\frac{\hbar^2}{2m} \frac{\partial^2}{\partial x^2} \Psi(x, t) + V(x, t) \Psi(x, t)$$

- 🌐 The equation will be solved on the time interval $[0, t_{\max}]$ with boundary conditions

$$\Psi(-x_{\max}, t) = \Psi(x_{\max}, t) = 0$$

and initial condition

$$\Psi(x, 0) = g(x),$$

where $g(x)$ is a specified function.

Numerical Approach

- The Crank Nicholson Approximation is:

$$i \frac{\psi_j^{n+1} - \psi_j^n}{\Delta t} = -\frac{1}{2} \left(\frac{\psi_{j+1}^{n+1} - 2\psi_j^{n+1} + \psi_{j-1}^{n+1}}{\Delta x^2} + \frac{\psi_{j+1}^n - 2\psi_j^n + \psi_{j-1}^n}{\Delta x^2} \right) + V_j \frac{1}{2} (\psi_j^{n+1} + \psi_j^n)$$

- This can be simplified (in conjunction with the boundary conditions) to a tridiagonal linear system, which can be solved using the Linear Algebra Pack (LAPACK) zgstv solver in MATLAB
- The solutions will then be visualized using some type of not yet researched visualization package to create MPEGs

Testing

- 🌐 The code will be tested through examining well known solved (closed-form) systems, such as the infinite square well
- 🌐 Furthermore, the conservation of probability law will be used to test the solution, namely:

$$I \equiv \int_0^1 \psi(x, t) \psi^*(x, t) dx = \text{constant.}$$

Which will be approximated by:

$$I \approx \sum_{j=1}^{nx-1} \psi_{j+1/2}^n \psi_{j+1/2}^{*n} \Delta x = \sum_{j=1}^{nx-1} \frac{1}{2} (\psi_j^n + \psi_{j+1}^n) \frac{1}{2} (\psi_j^{*n} + \psi_{j+1}^{*n}) \Delta x$$



Investigated Systems

- 🌐 The double delta well potential will be investigated as a crude model for bonding in Hydrogen gas. This will be examined with initial conditions in which the electrons are either in the excited ground state. The potential will be allowed to be time dependent to observe the effect of the electron's state on the bond length

Project Timeline

Dates	Activities
10/21 – 10/31	Derive equations for tridiagonal system. Research MATLAB functions and create outline for code structure.
11/01 – 11/15	Implement code
11/16 – 11/23	Test Code
11/23 – 11/27	Run experiments with variety of new potentials (both time-independent and dependent). Analyze Results.
11/28 – 11/30	Write Report
12/05	Deadline to Submit Report!!

References

-  <http://bho.phas.ubc.ca/210/Doc/term-projects/kdv.pdf>
-  <http://bho.phas.ubc.ca/210/Doc/term/schrodinger.pdf>



TRAFFIC SIMULATION MODEL USING CELLULAR AUTOMATA

PHYS 210

Term Project Proposal

Oscar Lo

OVERVIEW

Cellular Automaton:

- A grid consisting of unique cells (1D array)
- Cell interactions governed by distinct set of rules

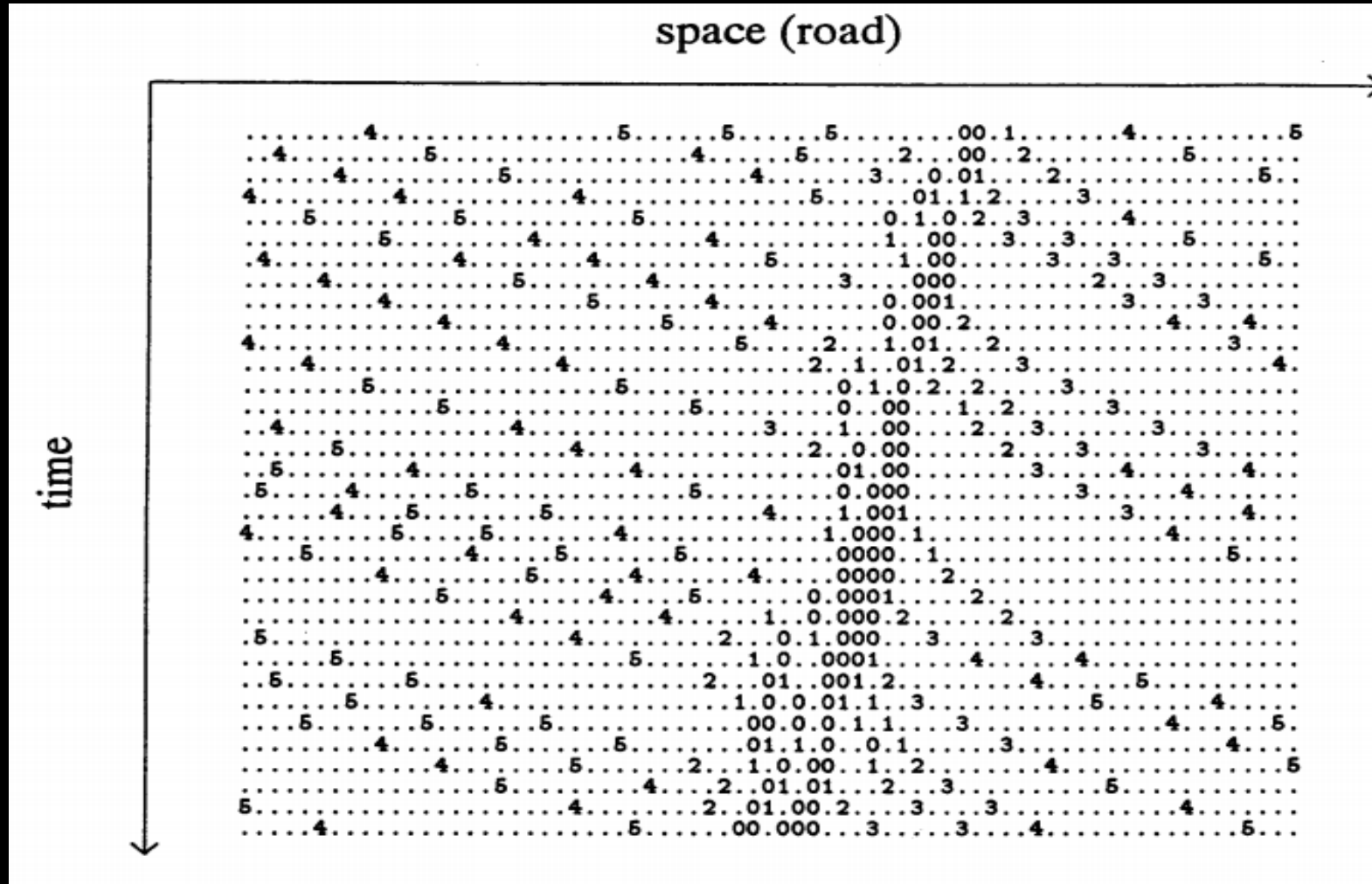
Traffic Simulation:

- Cells either full (occupied by car) or empty
- Full cells determine next action by analyzing adjacent cells

PROJECT GOALS

- Create a traffic simulation model using MATLAB
- Test this model using a variety of initial conditions
- Compare results to real-world observations
- Analyze causes of traffic congestion

SAMPLE OUTPUT



NUMERICAL APPROACH

- Rules of interaction based upon Nagel-Schreckenberg Model:

- 1. Acceleration:** if $v < v_{\max}$ and space between car ahead $> v_{\max} + 1$, $v \rightarrow v + 1$
- 2. Deceleration:** if car at position i spots car at position $i + j$ [$j \leq v$], $v \rightarrow j - 1$
- 3. Randomization:** velocity of moving car reduced by 1 with probability p : $v \rightarrow v - 1$
- 4. Motion:** cars advance v sites

MATHEMATICAL FORMULATION

- To simulate real conditions, density in a fixed region is averaged over a time period T:

$$\rho^T = \frac{1}{T} \sum_{t=t_0+1}^{t_0+T} n_i(t)$$

Where $n_i(t) = 1$ if site is occupied, 0 if empty

MATHEMATICAL FORMULATION PART 2

- q , the time-averaged traffic flow between points i and $i + 1$ is given by:

$$q^T = \frac{1}{T} \sum_{t=t_0+1}^{t_0+T} n_{i,i+1}(t)$$

$n_{i,i+1}(t) = 1$ if motion detected between i and $i + 1$, 0 otherwise

PROJECT TIMELINE

Dates	Activities
10/15 – 10/25	Preliminary research, derive equations, begin code design
10/26 – 11/16	Implement code
11/17 – 11/20	Test code
11/21 – 11/24	Run numerical experiments, analyze data, begin report
11/25 – 11/29	Finish report
11/29	Submit project (Deadline 12/03)

REFERENCES

- <http://laplace.physics.ubc.ca/phys210/Proposals-2012/L1A.pdf>
- Nagel and Schreckenberg, "A cellular automaton model for freeway traffic" J Phys I France 2 (1992) 2221-2229



**QUESTIONS?
COMMENTS?
SUGGESTIONS?**

The N-body Problem in Three Dimensions

Ben Martin

Physics 210 Term Project Proposal

October 21, 2014

Overview

- The N-body problem: in a system containing N massive particles with known initial positions and velocities, use the gravitational interactions between the objects to predict the motions of the particles for all future times
- Involves solving a 2nd order differential equation of acceleration, to obtain a position function

Project Goals

- Write a Matlab code giving numerical solutions to n-body differential equations, using 2nd order finite difference approximations
- To make an animated video of the interactions of n particles in a physical system involving only gravity
- Try using different initial conditions and parameter values to give different solutions
- To compare these results to known reference values, and use convergence tests, to check the accuracy of the model

Mathematical Formulation

- Vector equation for the total gravitational force on particle j with mass m_j and position \vec{r}_j :

$$\vec{F}_g = m_j \frac{d^2 \vec{r}_j}{dt^2} = \sum_{i=1, i \neq j}^n G \frac{m_j m_i}{|\vec{r}_i - \vec{r}_j|^2} \frac{(\vec{r}_i - \vec{r}_j)}{|\vec{r}_i - \vec{r}_j|}$$

- Divide out its mass to get its acceleration due to the total gravitational force:

$$\vec{a}_j(t) = \frac{d^2 \vec{r}_j}{dt^2} = \sum_{i=1, i \neq j}^n G \frac{m_i}{|\vec{r}_i - \vec{r}_j|^2} \frac{(\vec{r}_i - \vec{r}_j)}{|\vec{r}_i - \vec{r}_j|}$$

This is the 2nd order differential equation of a particle's acceleration, which must be solved to give the position function, $x(t)$.

But for n greater than 2, no explicit general solutions can be found! We can only approximate solutions.

Numerical Approach

- Simplify the problem by 'non-dimensionalizing' and multiplying by appropriate values to reduce constants to more manageable numbers (eg. making $G = 1$ and getting rid of units)
- **Use Finite Difference Approximation:** Convert the continuous domain \mathbf{t} (which can take on infinitely many values) into a discrete domain, called a **finite difference grid**, consisting of a finite number of \mathbf{t} values.
- Similarly, if we convert $\mathbf{r}(t)$ into the discretely valued approximation $f(t)$, then the 2nd order centred approximations of velocity and acceleration are:

$$\vec{v}(t) = f'(t) \approx \frac{f(t + \Delta t) - f(t - \Delta t)}{2\Delta t}$$

$$\vec{a}(t) = f''(t) \approx \frac{f(t + \Delta t) - 2f(t) + f(t - \Delta t)}{\Delta t^2}$$

$$\vec{a}(t) = f''(t) \approx \frac{f(t + \Delta t) - 2f(t) + f(t - \Delta t)}{\Delta t^2}$$

- Solve this $\mathbf{a}(\mathbf{t})$ equation for $\mathbf{f}(\mathbf{t} + \mathbf{delta} * \mathbf{t})$ to get a formula for a “step” in the finite difference approximation
- This is our approximation of the solution of differential equation, $\mathbf{r}(\mathbf{t})$

Visualization and Plotting

- Matlab will be used for plotting data for my report, and possibly for the animation (or I may use another program like xvs)

Testing and Numerical Experiments

- Try a variety of initial conditions and parameters to yield different solutions (change **x**, **v**, **a**, **delta t**, **m**, and **n**)
 - Use two particles of equal mass to test the code
 - Observe the effects of various masses, positions, velocities, and accelerations of the particles
- Compare results to known reference values to test the accuracy of the model
- Test accuracy using convergence tests

Timeline

Week	Plan
Oct 20	Finish deriving equations, start writing basic codes for simple situations
Oct 27	Write code for more complex situations (3D)
Nov 3	Test code
Nov 10	Choose good parameter values as inputs for the code, start plotting data & working on error/accuracy analysis
Nov 17	Start creating animations and finalizing numerical experiments, error analysis, etc.
Nov 24	Finish report and animations
Dec 1	Submit project by Dec 3

References

- <http://www.physics.buffalo.edu/phy410-505/topic5/>
- <http://laplace.physics.ubc.ca/210/Doc/fd/Nbody.pdf>
- http://en.wikipedia.org/wiki/N-body_problem

Questions?

Comments?

Phys 210 Term Project Proposal

Traffic Simulation by Cellular Automaton Model

Graham McIntosh

Overview

In a Cellular Automaton Model the behaviour of each cell at discrete time intervals is dictated by the behaviour of the cells around it and a touch of randomness.

A Cellular Automaton Model is a good approximation for traffic flow and is dictated by only a few simple rules.

Project Goals

1. Create a basic one-lane model of traffic flowing in a circle using MATLAB
2. Increase complexity of the model by simulating more real world elements such as traffic lights, cross-streets, double-lanes, etc.
3. Use a variety of different initial conditions to investigate behaviour
4. Test validity of the model against known solutions.

Nagel-Schreckenberg Model

1. Acceleration: if the velocity of the vehicle is lower than v_{\max} and if the distance to the next car ahead is larger than $v + 1$, the speed is advanced by one ($v = v + 1$)
2. Slowing down: if a vehicle at site i sees the next vehicle at site $i + j$ ($j \leq v$), it reduces its speed to $j - 1$ ($v = j - 1$)
3. Randomization: with probability p , the velocity of each vehicle(if greater than zero) is decreased by one ($v = v - 1$)
4. Car motion: each vehicle advanced v sites

Testing and Numerical Experiments

1. Test various initial conditions such as number of cars, speed limit, probability of decelerating etc.
2. Check that results match with the behaviour found in known models

Project Timeline

Dates	Activities
Oct. 20 – Oct. 27	Research, equations, code design
Oct. 28 – Nov. 14	Implement Code
Nov. 15 – Nov. 18	Test Code
Nov. 19 – Nov. 26	Run experiments, Analyze data, Start Report
Nov. 27 - Dec. 1	Finish Report
Dec. 2	Project Write-up submitted

References

K. Nagel, M. Schreckenberg. A cellular automaton model for freeway traffic. J. Phys I France 2 (1992): 2221-2229

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

http://en.wikipedia.org/wiki/Nagel-Schreckenberg_model



TOOMRE MODEL OF GALAXY COLLISIONS

Eric Neal

Physics 210 Term Project

OVERVIEW

- In the 1970's, brothers Alar and Juri Toomre conducted the first computer simulations of galaxies
 - Toomre's Model uses Newtonian mechanics to simulate the collisions of galaxies
- In Toomre's Model, stars are only acted upon by gravitational forces, they do not contribute to gravitational interactions
 - They are treated almost as test particles
 - In the (rare) case of two stars actually colliding, it will be ignored and the stars will appear to pass through each other
- My simulation will be in 2 dimensions, in order to make the simulation simpler and easier to carry out

PROJECT GOALS

- To write MATLAB code which simulates the collision of 2 galaxies using the Toomre Model
- To explore these collisions over a range of initial conditions
 - I will experiment by varying initial positions, velocities, masses, and shapes of the 2 colliding galaxies
- If I have time, I would like to simulate the collision of the Milky Way and Andromeda galaxies
- To establish the correctness of my simulations using the law of conservation of energy and by comparing with existing simulations and observations

MATHEMATICAL FORMULAS

- In the Toomre model, Newtonian mechanics are sufficient for our approximation
- I will assume that the only force acting on the galaxies is gravity, all other forces will be considered negligible
 - I will also assume that there are no gravitational forces from anything outside the system, such as other galaxies in the distance

MATHEMATICAL FORMULAS CONT'D

- Newton's Law of Gravitation

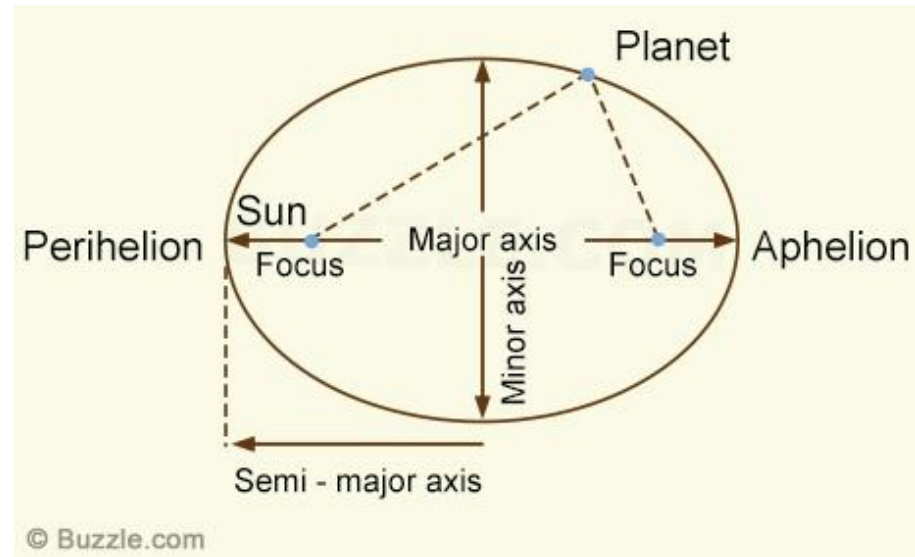
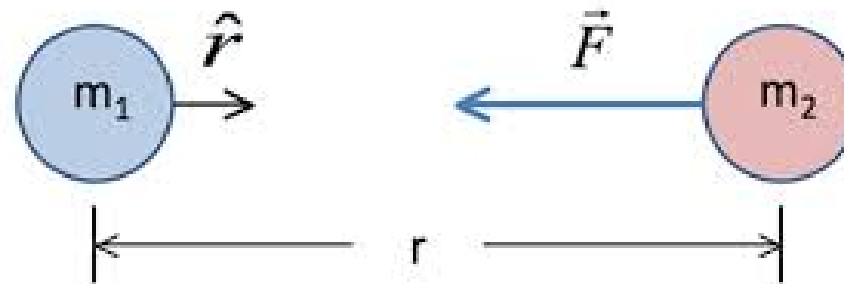
$$\vec{F} = \frac{G m_1 m_2}{r^2} \hat{r}$$

- Using Newton's second law ($F = ma$)

$$a = \sum \frac{F}{m} = \sum \frac{Gm}{r^2}$$

- Kepler's third law

$$P^2 = \frac{4\pi^2}{Gm} a^3$$



NUMERICAL APPROACH

- Derive the equations of motion from Newton's second law of motion

$$\bullet a = \frac{F}{m} = \frac{\partial v}{\partial t} = \frac{\partial^2 r}{\partial t^2}$$

- From the equations of motion, I will derive and use these FDAs

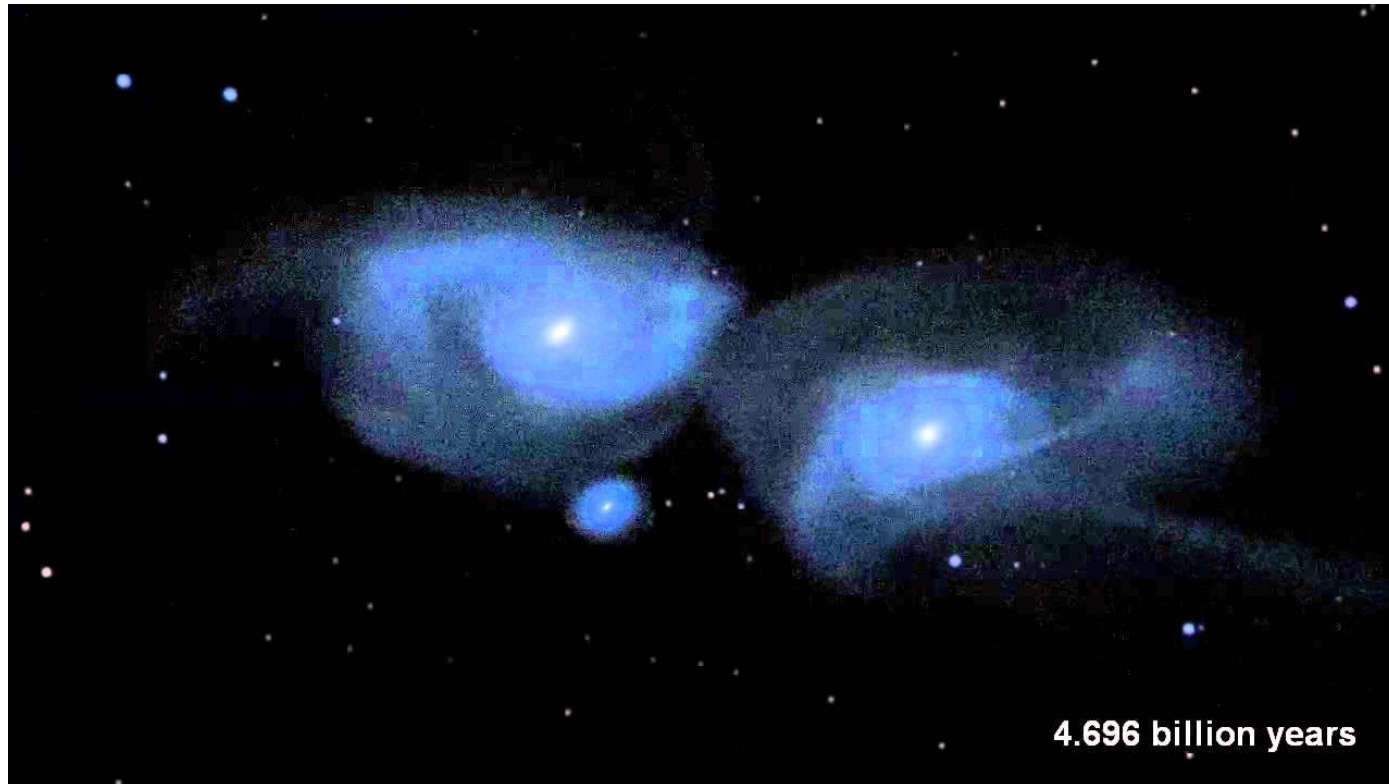
$$\bullet a = \frac{F}{m} = \frac{\partial v}{\partial t} \approx \frac{(v_{i+1,j} - v_{i-1,j})}{2\Delta t}$$

$$\bullet a = \frac{F}{m} = \frac{\partial^2 r}{\partial t^2} \approx \frac{(r_{j+1,j} - 2r_{j,j} + r_{j-1,j})}{\Delta t^2}$$

- Define initial conditions
- Implement over finite domains of space and time

VISUALIZATION AND PLOTTING TOOLS

- I will use MATLAB to create an mpeg file of my simulation



TESTING AND NUMERICAL EXPERIMENTS

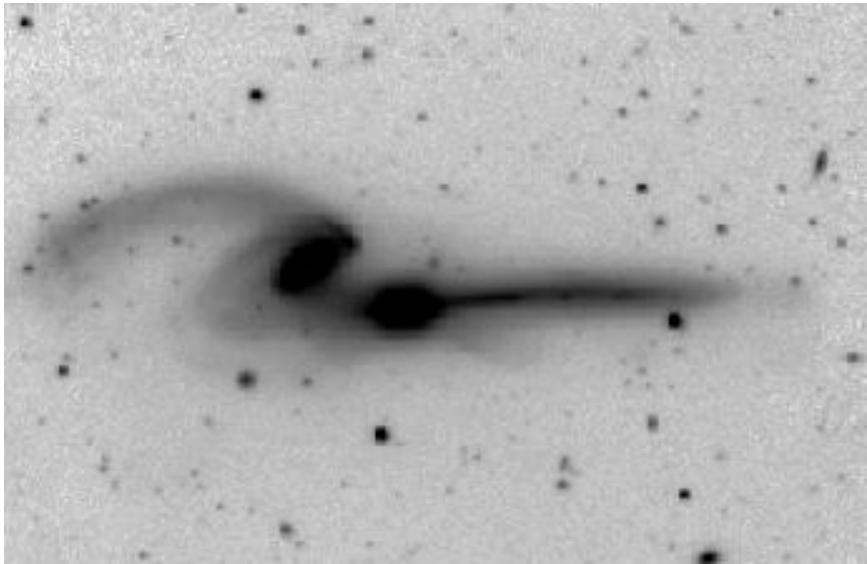
- Using the collision of 2 galaxies, test over a wide range of conditions
 - Investigate a variety of different collision situations
- Try to find a good balance between step size and computation time
 - ($n \approx 1000?$)
- Check my results using conservation of energy and existing simulations and observations
- If I have time, investigate the specific collision of our Milky Way with the Andromeda Galaxy
 - Set initial conditions to our predictions of how this collision will occur
 - Possibly investigate what will happen to our solar system in particular during the collision

PROJECT TIMELINE

Dates	Goals
Oct 20 – Nov 2	Basic research, derive equations, begin designing code
Nov 3 – 14	Implement code
Nov 15 -21	Test code
Nov 22 - 29	Run numerical experiments, analyze data, begin report
Nov 29 - 30	Finish Report
Dec 1 - 3	Finishing touches and submit project

REFERENCES


- <http://laplace.physics.ubc.ca/210/Term.html>
- <http://faculty.etsu.edu/smithbj/collisions/collisions.html>
- http://en.wikipedia.org/wiki/Alar_Toomre



The background features abstract, flowing waves of color. At the top, a thick, vibrant magenta wave curves across the frame. Below it, a thinner, lighter blue wave flows. At the bottom, another thick magenta wave curves upwards, overlapping with a blue wave that flows from the right side towards the center. The overall effect is a sense of dynamic movement and modern design.

THE END

Questions?



Gravitational N-Body Problem: Simulation of Orbiting Planets and Binary Stars

Physics 210 L1A

Kenneth (Pin-Cheng) Pan

37219136

Overview

- In classical physics, every matter with mass are subjected to be influenced by the force of gravity.
- Newton's universal gravitational law describes how two or more objects interact under the force of gravity.
- Kepler's laws of planetary motion, specifically addresses the motion of larger bodies (e.g. planets and stars)
- The force gravity is dependent on mass of objects and the distance between the two objects.

Project Goals

- Write an MATLAB (octave) code which solves the Newton's universal gravitational equation numerically, using second-order finite difference approximation techniques
- Use the code above to create a visual representation and simulation of the interacting particles or planets
- Compare the simulation generated with the known solution to establish correctness of the implementation of the code
- Alter initial conditions and parameters such as mass and distance to see how each effect the planetary and stellar motion of the binary star system

Mathematical Equations: Newton's Laws of Universal Gravitational Force

- Newton's 2nd Law of Motion:

$$\checkmark F = m \frac{d^2x}{dt^2} = m \frac{dv}{dt} = ma$$

- Centripital Force:

$$\checkmark F_c = ma_c = m \frac{v^2}{r} = m \frac{4\pi^2 r}{P^2}$$

- Law of Universal Gravity

$$\checkmark 2 \text{ bodies: } F_{12} = \frac{Gm_1m_2}{|r_{12}|^2} \hat{r}_{12}$$

$$\checkmark \text{Multi-bodies: } F_g = - \sum_{k=1, i \neq k}^{N-1} \frac{Gm_k m_i}{|r_{ki}|^3} r_{ki}$$

Mathematical Equations: Kepler's Law of Planetary Motion, Energies and Angular Momentum

- Angular Momentum:

$$\blacktriangleright \vec{L} = m\vec{r} \times \vec{v}$$

- Kinetic & Potential Energies:

$$\blacktriangleright K = \frac{1}{2}mv^2, U = -\frac{GMm}{r}$$

- Kepler 2nd Law of Planetary Motion (Area v.s. angular velocity):

$$\blacktriangleright \partial A = \frac{1}{2}r^2 \frac{\partial \theta}{\partial t} dt$$

- Kepler 3rd Law of Planetary Motion (Period v.s. Semi-major Axis):

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

Numerical Approach

- Newton's laws will be discretized using second order finite difference approximation technique where the continuum domain of time and space (x,t) will be replaced with lattice (discrete grids) of points (x_i, t_n) such that:

$$\begin{aligned}x_i &= -x_{max} + i\Delta x & i &= 1,2,3 \dots n_x \\t_n &= n\Delta t & n &= 0,1,2 \dots n_t\end{aligned}$$

- Newton's second law of motion will then will look like:

$$\begin{aligned}F &= \sum_{i=1, i \neq j}^n F_{ij} = m \frac{d^2 x_i}{dt_i^2} - m \frac{d^2 x_j}{dt_j^2} \\ &= m \frac{d^2 (x_i - x_j)}{d(t_i - t_j)^2}\end{aligned}$$

Visualization and Plotting Tools

- I will use xvs for interactive analysis and generation of mpeg animations, and
- MATLAB's plotting facilities for plots to be included in my report

Testing and Numerical Experiments

- Testing

- Convergence Testing: Fixed the initial condition. Let $\Delta x = h = \Delta t / \lambda$, use this discretization factor to form discretization scale $h/2, h/4, h/8 \dots$ Ensure to include $O(h^2)$ convergence behavior.

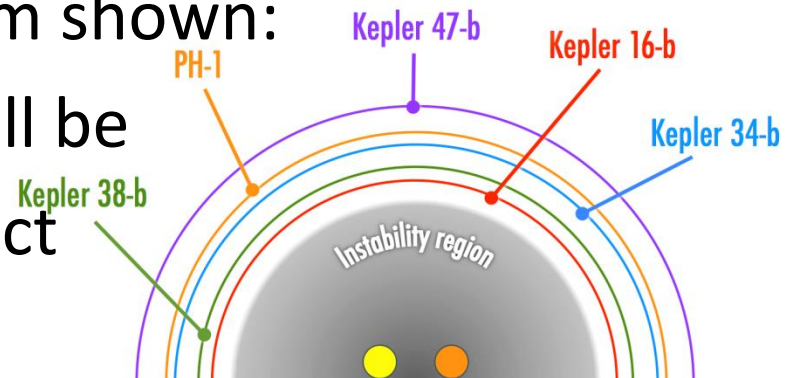
- Numerical Experiment

- Investigate the planetary and stellar motion of a binary star system

Observed circumbinary planets
(orbits normalized to the instability region)

- Replicate binary star system shown:

- Mass, distance, velocity will be varied to investigate the effect



Project Timeline

Dates	Tasks
10/13 – 10/24	Basic Researches and Proposal
10/25 – 11/15	Implement Code
11/16 – 11/19	Test Code
11/20 – 11/25	Run Numerical Experiment, Analyze Data, Begin Report
11/26 – 11/28	Finish Report
11/28	Submit Report

References

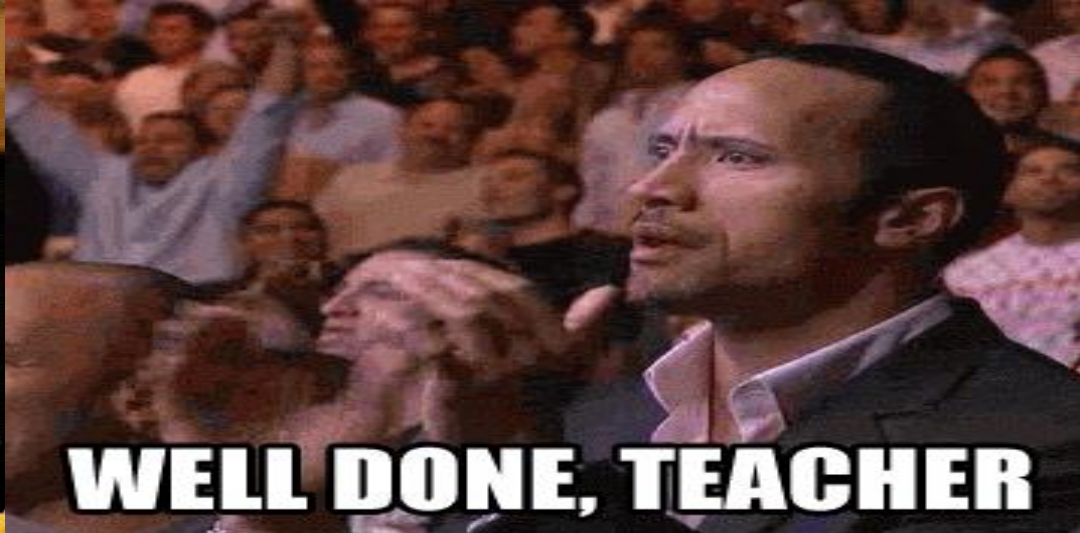
- ASTR 200 Frontiers of Astrophysics Lecture Notes by Dr. Paul Hickson:
 - http://www.phas.ubc.ca/~hickson/astr200/astr200_2013_notes.pdf
- Edmund Bertschinger (1998). "[Simulations of structure formation in the universe](#)". *Annual Review of Astronomy and Astrophysics* **36** (1): 599–654. [Bibcode:1998ARA&A..36..599B](#). [doi:10.1146/annurev.astro.36.1.599](#)
- PHYS 210 Introduction to Computational Physics Finite Difference Solution of N-Body Problems:
 - <http://laplace.phas.ubc.ca/210/Doc/fd/nbody.pdf>



GIFSec.com



Snape Approves



Well Done, Teacher



Very Good

I Clap for You



He Deserves a Clap



Needs More Clapping

quickmeme.com

TROLL.ME.GIF

quickmeme.com

A Simple Model of Neural Networks

Physics 210 Term Project Presentation

Fall 2014

Kristen Ruhnke

Overview

- A human brain is made up of many neurons (10^{12}) which form a neural network
- They are connected by axons and dendrites
- Neurons highly interconnected (around 10^4 inputs and outputs)

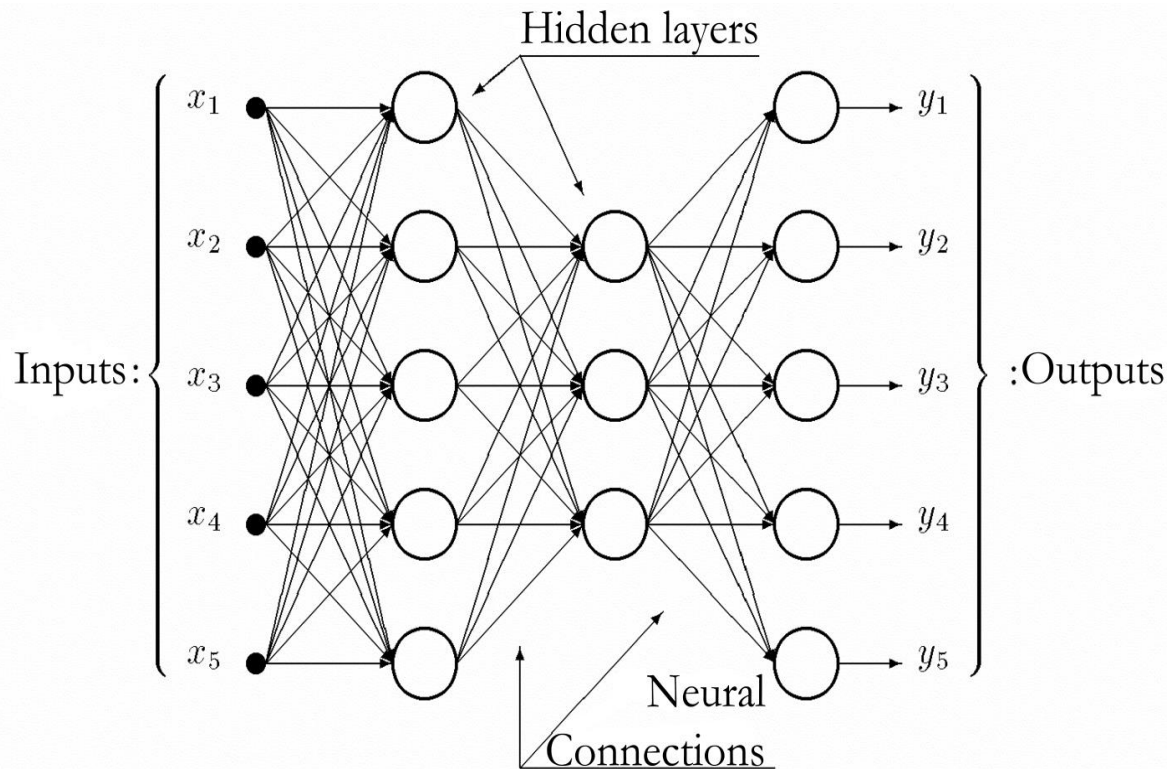


Figure 1

- Neurons communicate through electrical signals
- Firing usually order 10^{-3} s and 5×10^{-2} V
- Firing rate dependent on firing rates of inputting neurons
- Individual neurons fire at rates determined by the effective fields established by interacting with other neurons

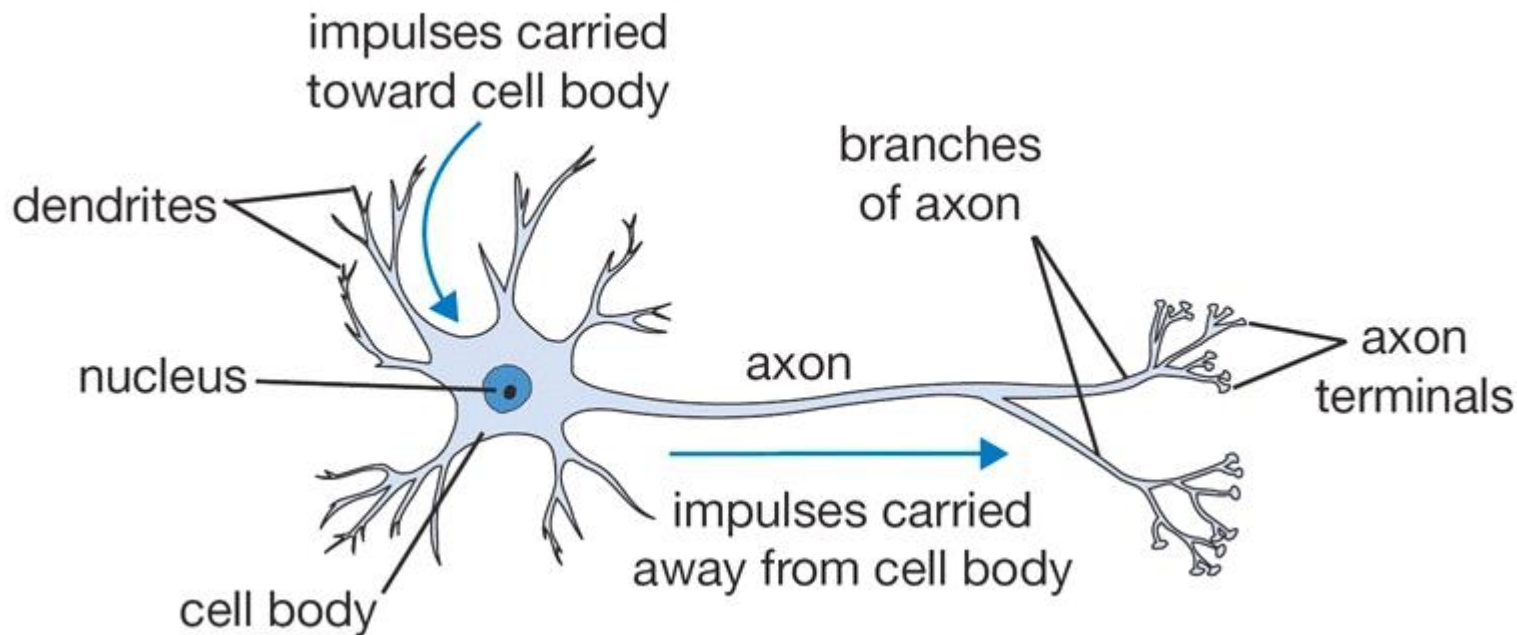


Figure 2

- Effective energies of neurons thought of in terms of spin (- or +)
- Can be modeled by Ising spin model and using the Monte Carlo method (assuming symmetry)
- The energy of a network at a specific time depends on the energy landscape or spin configuration
- Each stored pattern will correspond to a minimum in the energy

Project Goals

- Write a code in Matlab that will correctly model a simple neural network using the Ising model
- Explore the effect of damage to the neural network on memory, specifically recognizing patterns
- Test how many patterns can be stored and how different patterns effect the system
- Also possibly use this model to explore the capability to learn new patterns

Mathematical Formulation

The effective energies can be represented by the formula

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

Where $J_{i,j}$ is the interaction energy, and s_i and s_j are spins i and j
A convenient choice for the interaction energies is

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

Where M is the number of stored patterns, m is the pattern, and $J_{i,j}$ is the effect of neuron i on neuron j

Numerical Approach

- The firing rate of a neuron can be assumed to only have two states, $s = \pm 1$. Thus the Ising spin model can be used to represent this system
- Using the Monte Carlo method which by finding E_{flip} and looking at the sign, changes the state of the neural network to something with equal or lower energy
- This method may need to be applied iteratively

Visualization and Plotting Tools

- I will create lattices with matlab's matrix functionality

Testing and Numerical Experiments

- I will test this by changing the pattern given to the system
- I will test the amount of patterns that can be stored
- I will introduce damage to the interactions between neurons by randomly setting $J_{i,j}$ to 0 in the matrix

Project Timeline

Dates:	Activities:
10/20- 10/30	Do preliminary research, design, begin coding
10/30- 11/15	Write code
11/16- 11/20	Test code
11/21- 11/26	Run experiments, analyze data
11/27- 11/30	Finish report and hand in.

References

- <http://laplace.physics.ubc.ca/210/Doc/term/Giordano-12.3-4-Neural-Networks.pdf>
- http://en.wikipedia.org/wiki/Monte_Carlo_method
- <http://farside.ph.utexas.edu/teaching/329/lectures/node110.html>
- figure
1: <http://bulyaki.files.wordpress.com/2013/02/ffnetwork.png>
- figure
2: <http://www.wpclipart.com/medical/anatomy/cells/neuron/neuron.png>



Finite Difference solution of a N-body Gravitational problem

PHYS210 TERM PROJECT PROPOSAL

MAGALIE TATISCHEFF 55376131

Overview

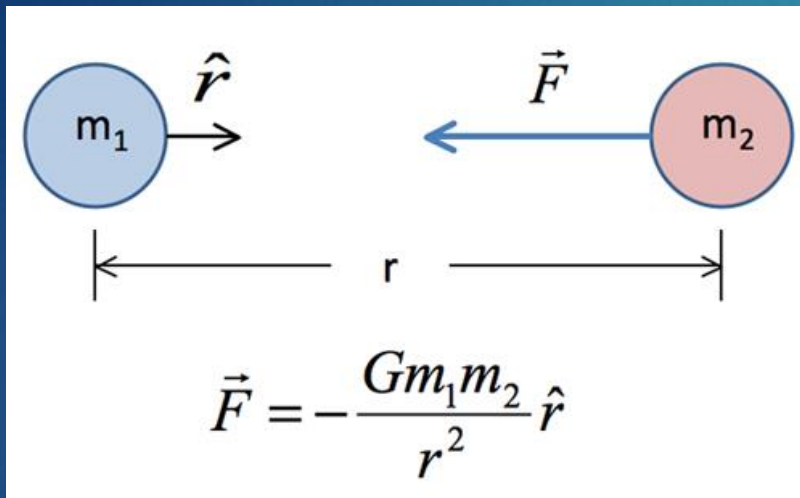
- ▶ The motion of many particles can be predicted with
 - ▶ initial conditions
 - ▶ Newton's second law
 - ▶ Newton's gravitational law
- ▶ By using FDA we can run a simulation on a computer of how N-particles interact gravitationally.

Project Goals

- ▶ Code on MATLAB and use FDA to solve N-body gravitational problem
- ▶ Be able to visualize the solution through animation
- ▶ Test, experiment and compare to acquired knowledge



Mathematical formula



$$F = dp/dt$$

Numerical approach

- ▶ The equations will be approximated using FDA.

Testing



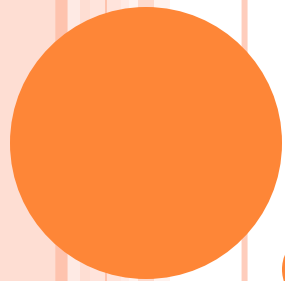
- ▶ Usage of different:
 - ▶ Initial conditions
 - ▶ Discretization scales
- ▶ Check if results seem logical

Project Timeline



References

- ▶ <http://laplace.physics.ubc.ca/210/Doc/term-projects/kdv.pdf>
- ▶ http://en.wikipedia.org/wiki/Weakly_interacting_massive_particles
- ▶ <http://appel.nasa.gov/2011/02/02/the-path-to-scientific-discoveries-designing-the-cassini-solstice-mission-trajectory/>
- ▶ <http://www.learner.org/courses/physics/unit/text.html?unit=3&secNum=3>



CHUA'S CIRCUIT

Kaitlin Thachuk

**Phys 210 Project Proposal
October 2014**

OVERVIEW

- Chua's circuit was the first physical implementation specially designed to exhibit chaos
- Circuit composed of 4 linear elements: an inductor (L), a resistor (R), and two Capacitors (C_1 and C_2) and one locally active nonlinear element usually a Chua diode which functions as a locally active resistor

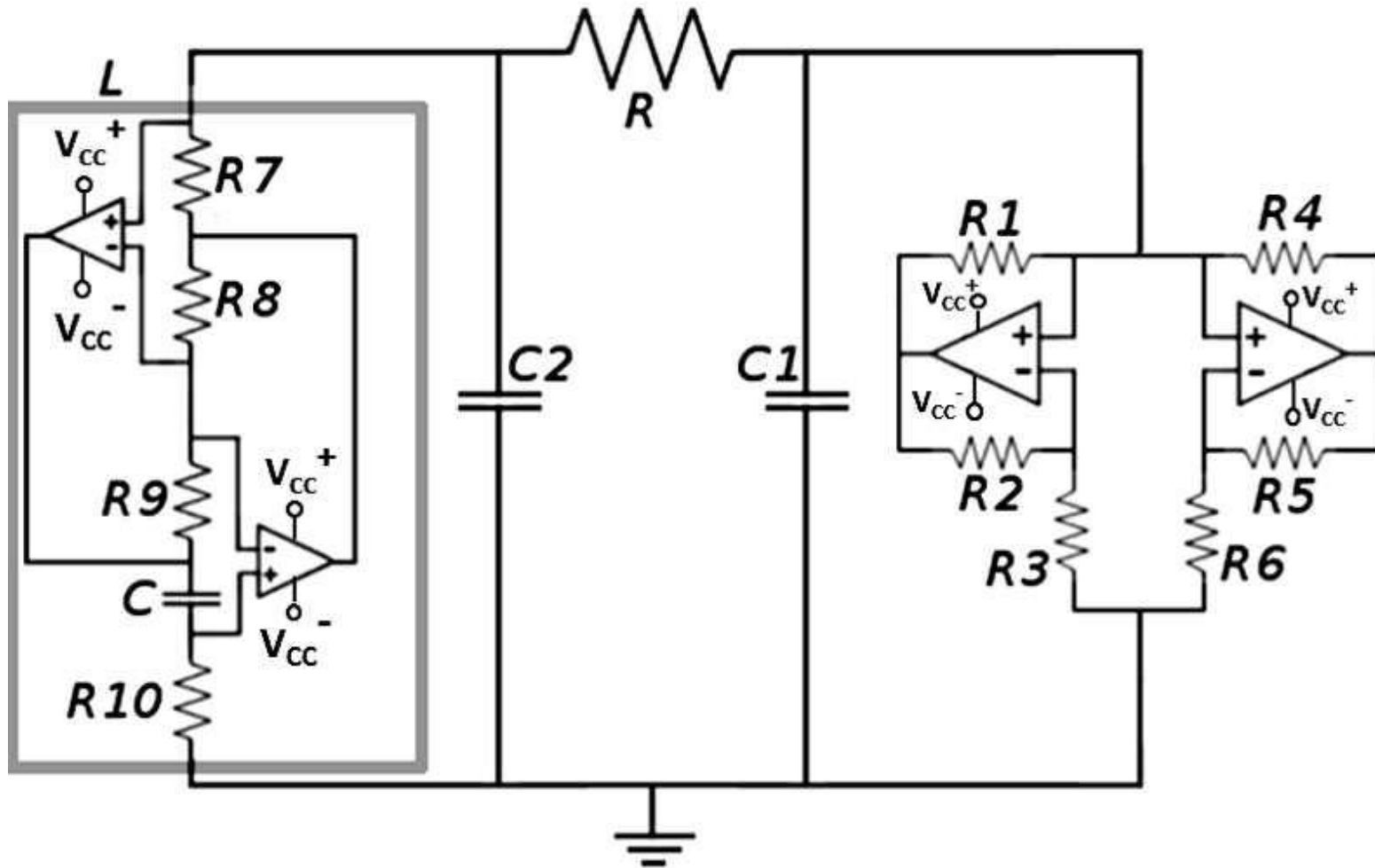


GOALS

- To use MATLAB (octave) to create a program that solves the non-linear ODEs associated with Chua's circuit
- To use this program to test for interesting chaotic properties such as sensitivity to initial parameters, strange attractors, and period doubling
- To compare the values produced with known solutions



DIAGRAM OF ONE VERSION OF CHUA'S CIRCUIT



MATHEMATICAL EQUATIONS

- Chua's nonlinear ODEs are given by:

$$\frac{dx}{dy} = \alpha(y - x - g(x))$$

$$\frac{dy}{dt} = x - y + z$$

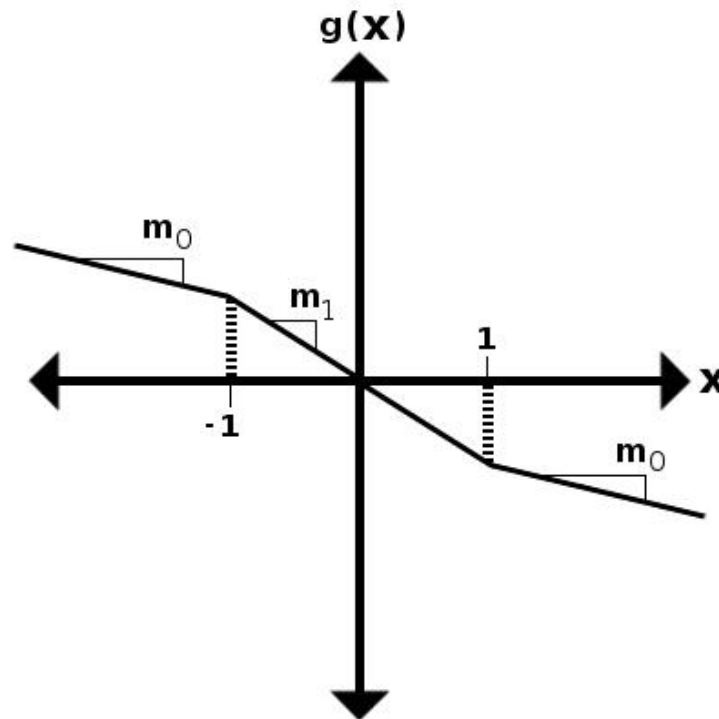
$$\frac{dz}{dt} = -\beta y$$

$$g(x) = m_1 x + \frac{1}{2}(m_0 - m_1)(|x + 1| - |x - 1|)$$



MATHEMATICAL EQUATIONS CONTINUED

- Looking at the diagram of change in resistance vs current across the diode we see that m_0 and m_1 must be negative

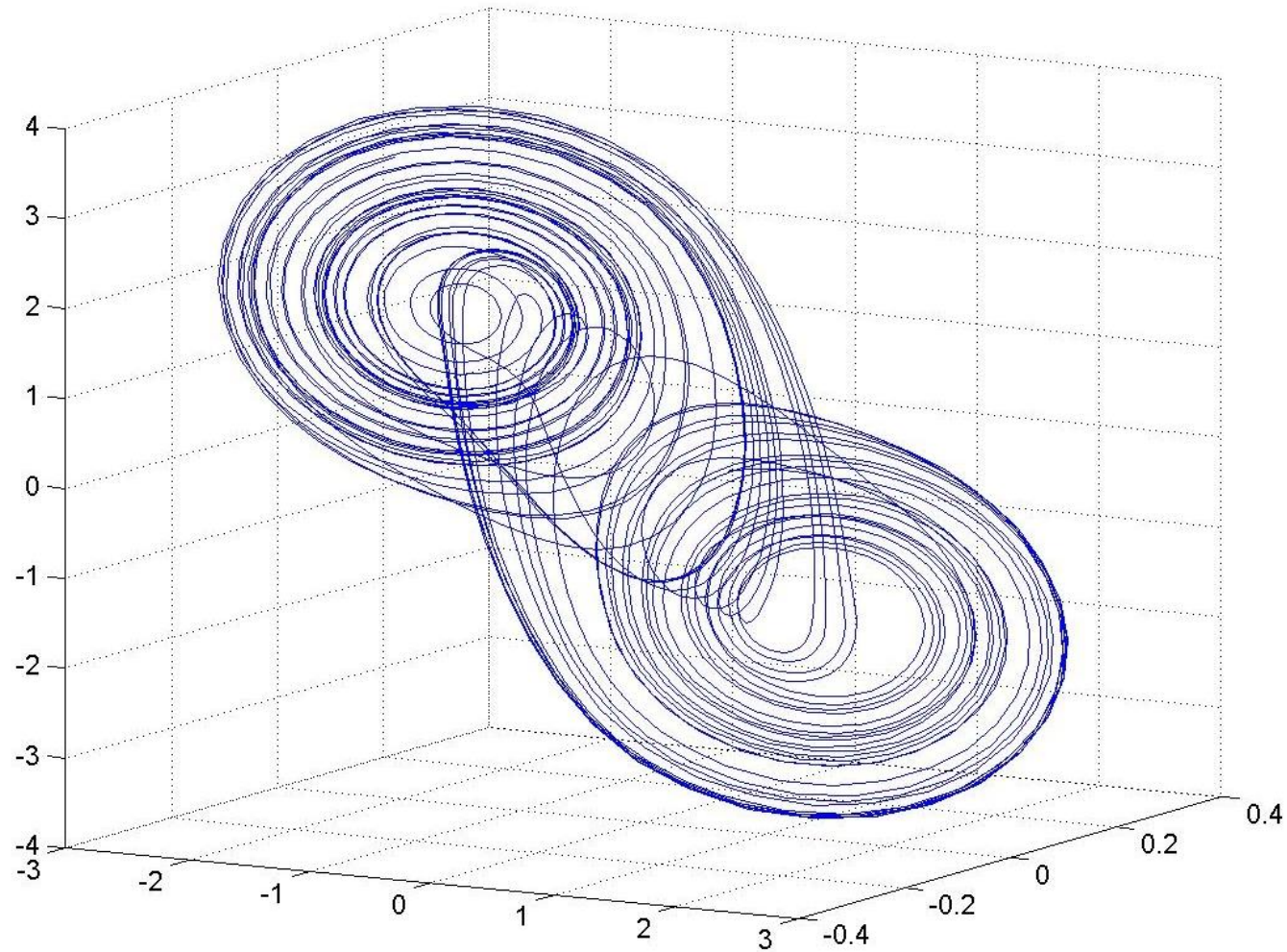


TESTING & NUMERICAL TECHNIQUES

- Not too sure what numerical techniques I will be using outside of MATLAB's built in ODE solver
- Whichever method I use I will test for accuracy and compare against known solutions
- I will find solutions to the points of interest: period doubling and the range that produces the strange attractor, as well as explore the sensitivity of the system to initial parameters



DOUBLE SCROLL ATTRACTOR



TIMELINE

DATES	ACTIVITES
Oct. 27- Oct. 31	Design and begin running code
Nov. 3- Nov7	Run and test code
Nov.10- Nov.14	Conduct experiments and begin writing report
Nov. 17- Nov. 21	Analyze results and continue writing report
Nov. 24-Nov. 28	Polish up code and report
Dec. 3	Hand in report



REFERENCES

- Image on slide four:
<http://www.chuacircuits.com/howtobuild2.php>
- Image on slide 7: "Double scroll attractor from Matlab simulation" by www.chuacircuits.com
- Image on slide 6:
<http://www.chuacircuits.com/diagram.php>
- <http://www.chuacircuits.com/>
- http://www.scholarpedia.org/article/Chua_circuit
- http://www.cmp.caltech.edu/~mcc/chaos_new/Chua.html

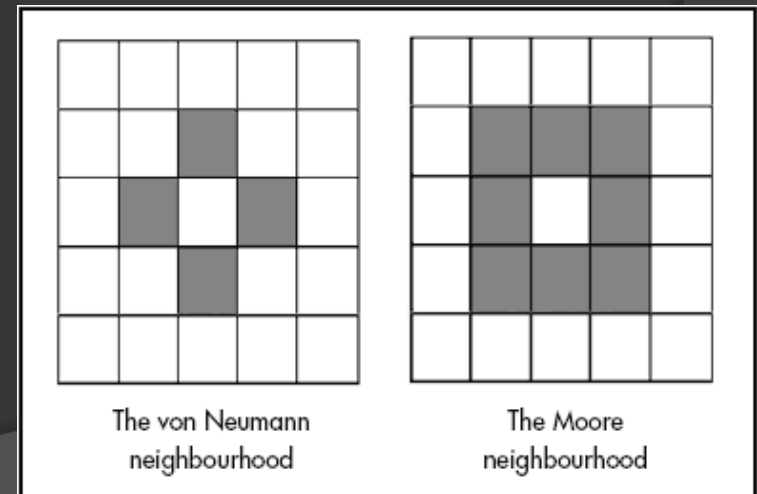
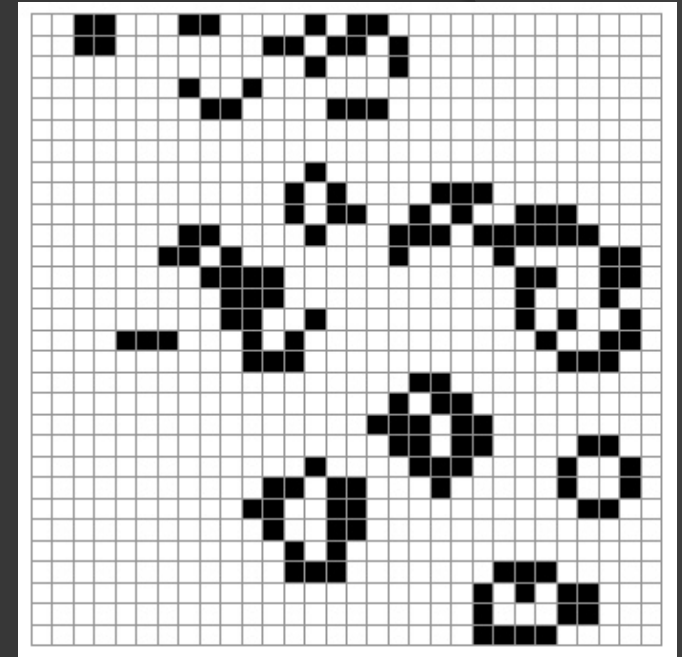


CELLULAR AUTOMATA: MODELING FOREST FIRES

CARSON VANDER NOOT - 2014

Overview

- Model forest fires and regrowth using cellular automata
 - Grid of finite size
 - Set # of states
 - State of neighbour influences cell
- Flat surface
- von Neumann neighbourhood



Goals

- ◎ Simulate burning and regrowth of forest using 2D cellular automata in MATLAB
- ◎ Create most realistic model as possible
 - Adjust rules and parameters to best recreate natural fires

Rules and Equations

- ⊙ 3 states: empty, alive, burning

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

- ⊙ empty \rightarrow empty

- ⊙ alive \rightarrow alive

 - \rightarrow burning if neighbour is burning

- ⊙ burning \rightarrow empty

More Equations

◎ $S_c^{t+1} = f(S_c^t, S_{n(c)}^t)$

◎ $n(c)$

- von Neumann neighbourhood

	$c_{i,j-1}$	
$c_{i-1,j}$	$c_{i,j}$	$c_{i+1,j}$
	$c_{i,j+1}$	

◎ What is f ?

- If any of $S_{n(c)}^t$ are burning and c is alive, then c is now burning
- Fancy equations named after people (Frandsen, Rothermel... etc.)

More Rules

- ⊙ Spontaneous life/fire
- ⊙ Empty cells have probability “ a ” of becoming alive
- ⊙ Alive cells have probability “ b ” of spontaneously burning
 - $a \ll b \ll T_{burn}$
 - b/a = average trees created between fires

Testing and Analysis

- ◎ Graphs → power-law relation for non-cumulative frequency-size distribution
- ◎ Rules
 - What f is
 - Burn time
 - Spontaneous fire/life probabilities, $a + b$
- ◎ Environment
 - Square vs. hexagonal grid
 - Wind
 - Water
 - Toroid loop

Timeline

- ⦿ Oct: 20-30: Research, understand equations
- ⦿ Nov: 1-11: Design and Implement code
- ⦿ Nov: 12-15: Testing
- ⦿ Nov: 16-24: Run experiment, analysis
- ⦿ Nov: 24-29: Write report
- ⦿ Nov: 30: submit

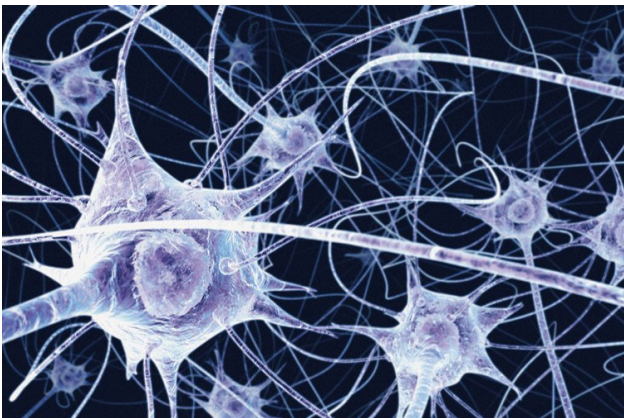
References

- ⦿ <https://courses.cit.cornell.edu/bionb441/CA/>
- ⦿ http://en.wikipedia.org/wiki/Forest-fire_model
- ⦿ <http://www.fundacioent.cat/images/stories/ENT/articles/automata.pdf>
- ⦿ <http://www.sciencedirect.com/science/article/pii/S0307904X06000916>

Neural Networks as an Active Computer Memory

Phys 210 Term Project Proposal

Kevin Wang



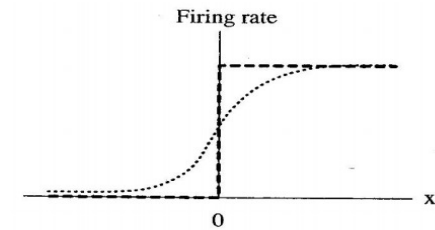
Overveiw

- Neural networks consists of individual "neurons" that are connected to each other; each firing signals at different periodic rates. Rate is a function of inputted signals
- Human brain is a network consisting of approx 10^{12} neurons; each with 10^4 connections

Goals

- Map out a grid of neurons using Matlab and allow them to change states as signaling occurs
- Manipulate relations between neurons such that the network converges on set patterns
- Implement multiple patterns and pattern learning
- Explore different rate relationships

Mathematical Implementation



- Actual rate function is non-linear but we approximate the neuron into 2 states (on and off; -1 and 1)
- Each of the neurons are connected to each other. A memory is chosen based on a theoretical energy value of the set.

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

- Where $s(i)$ and $s(j)$ represent the states of the 2 connected neurons and J contains the desired memory state such that E is most negative
- An example J is $J = s(i, m) * s(j, m)$ where m is the pattern we want in the memory. If the set = memory, each term cancels to 1 and E is most negative

- To implement more than one pattern:

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

- Implementing learning and forgetting:

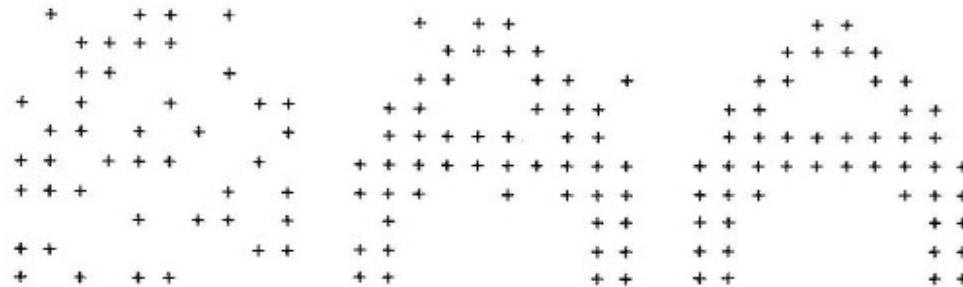
$$J_{i,j}(\text{new}) = \beta J_{i,j}(\text{old}) + \alpha s_i(p) s_j(p) ,$$

Numerical Approach

- All neurons are represented as a 1D array.
- The matrix J must list all neuron connections, for each neuron that is n^2 connections, therefore for a network of 100 neurons, there must be 100^2 J values.
- A neuron flip if it lowers total energy in network

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

- In one time step, the neurons that need to be flipped are determined and then the flips are implemented



Visualization, Testing, and Experimentation

- I will be working with 1 dimensional arrays to represent the network and as such, the array just needs to be displayed for each pass of the neuron flips.
- Convergence should occur for less than 10% of values randomized from the memory shape
- I will test for the amount of acceptable deviation from the memory that will still allow the network to converge to the same shape
- This test will be extrapolated to networks with multiple memory configurations; who's energy distance can be calculated with:

$$\Delta_{m,n} = \frac{1}{N} \sum_i [s_i(m) - s_i(n)]^2 ,$$

- Tests for memory degradation where a percentage of J is set to the null value of 0.

Project Timeline

Date	Activity
10/13 – 10/24	More research, develop equations, model diststructure of code
10/25 – 11/15	Implement code
11/16 – 11/19	Testing and debugging
11/20 – 11/25	Run experiments, analyze data, start report
11/26 – 11/28	Finish report
11/28	Project Submission

- References : <http://laplace.physics.ubc.ca/210/Doc/term/Giordano-12.3-4-Neural-Networks.pdf>

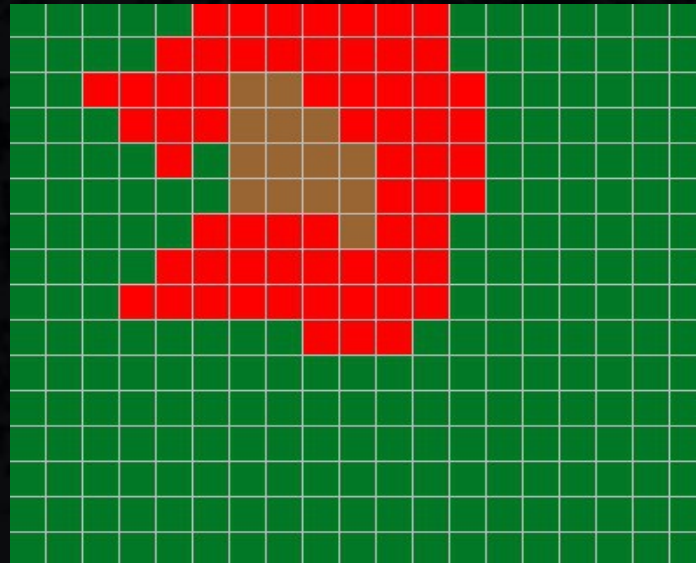
Modelling Forest Fires using Cellular Automata

Phys 210 Term Project Proposal

Cliff Wells

Overview

- A Forest will be represented with a large grid of cells in matlab



- Each cell in the grid can be in one of three states: Forest, Fire, or Empty
- These cells can change states based on rules

Rules of the Model

- Time will progress in discrete “ticks”
- For each “tick” there is a **CHANCE** for each cell to change based on the following rules:

Ignition (Pi): Forest changes to Fire when adjacent to Fire (High)

Extinguishing (Pe): Fire changes to Empty spontaneously (Medium)

Growth (Pg): Empty changes to Forest when adjacent to Forest (Low)

Lightning (Pl): Forest changes to Fire spontaneously (Very Low)

Project Goals / Research

- In my system the following parameters can be altered:
 - Event probabilities (P_i , P_l , P_e , P_g)
 - The size of the grid
- How do these parameters effect:
 - Evolution trends
 - Long term outcomes
 - Stable population size

Project Timeline

- Nov 12th, Implement code in matlab
- Nov 15th, Debug code
- Nov 15-20th, Play with parameters
- Nov 20-28th, Analyze and report on interesting relationships
- Nov 30th, Finish and submit project

References

- http://en.wikipedia.org/wiki/Forest-fire_model
- <https://www.youtube.com/watch?v=RVcya3Tg5QU>

THE GRAVITATIONAL INTERACTION N- BODY PARTICLES

Physics 210 Term Project
Proposal
By: Brendon Wong

PROJECT OVERVIEW

The N- body problem predicts the motion of objects interacting due to gravitational forces

Used to simulate the future motion of objects, given various initial conditions such as mass, position, velocity

Used to aid in understanding the dynamics of various celestial bodies (Sun, moon, earth interactions)

PROJECT GOALS

To write a *MATLAB*/*Octave* code to solve the n-body problem

To simulate n-body motion and interactions in 2D or 3D with FDAs

To test various initial conditions with n particles and study the results

MATHEMATICAL FORMULATION

Newton's Law of Universal Gravitation

$$\vec{F} = -G \frac{m_1 m_2}{|r_2 - r_1|} \hat{r}_{12}$$

Newton's Second Law of Motion

$$F = ma$$

More mathematical formulations to be used as needed

NUMERICAL APPROACH

Problem will be solved with finite differencing approximations

Small time intervals will be used for clearer motion with each step

May have to modify some equations or introduce another factor/condition for when distance between objects go 0

VISUALIZATION AND PLOTTING TOOLS

Will assume MATLAB plotting and visualization tools are sufficient for producing plots and visual simulations

If insufficient, will look for another visualization tool to generate animations for project

TESTING AND NUMERICAL EXPERIMENTS

Test and observe various initial conditions

Test with varying number of bodies

Compare with known solutions and n-body systems

Check agreement with conservation laws

More tests and investigations to be executed as needed

PROJECT TIMELINE

Dates	Activities
Oct 20 th – Oct 27 th	Begin research, study and derive necessary equations, design code
Oct 28 th – Nov 3 rd	Write and implement code
Nov 4 nd – Nov 7 th	Test Code
Nov 8 th – Nov 13 th	Run numerical experiments and analysis, begin writing report
Nov 14 th – 26 th	Complete and finalize report
Nov 27 th	Submit project

REFERENCES

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

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

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

<http://laplace.physics.ubc.ca/phys210/> (Referencing structures of presentations from previous years and instructor demo presentation)



Simulation of forest fire

Haonan Wu

10/20/2014

Overview

- ▶ Simulation of forest fires using automata simulation
- ▶ For simplicity, there will be no consideration of the wind.
- ▶ Whether a cell is on fire or not depends on the states of its neighbours.



Project Goals

- ▶ Create a simulation by writing codes on MATLAB
- ▶ Test if the behavior of the model roughly states the real behavior of forest fires
- ▶ Enhance proficiency of Matla and cellular automata



Mathematic Formulation

► Transition rule:

$$f \in F$$

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

S_c^{t+1} : Condition of new cell

S_c^t : condition of current cell

S_{ncc}^t : condition of neighbouring cell



Testing and Numerical Experiments

- ▶ Using square cellular automata, start with one origin
- ▶ Determine the time that takes the source spreads within the whole region
- ▶ Attempt to find out the time difference of putting the source at the center of the region or on the edge
- ▶ Develop further about multiple fire sources



Project Timeline

Date	Task
10/22-10/30	Do basic research, derive equations and begin coding
10/30-11/10	Implement code
11/10-11/15	Test code
11/15-11/23	Run numerical experiment, analyze data, begin report
11/23-11/29	Finish report
11/30	Submit (deadline 12/2)

Predicting Stock Prices

Using Artificial Neural Networks

Overview & Benefits



Quantitative analysis is a major source of employment for people with mathematics and physics Ph.D. degrees!

How to predict stock prices?

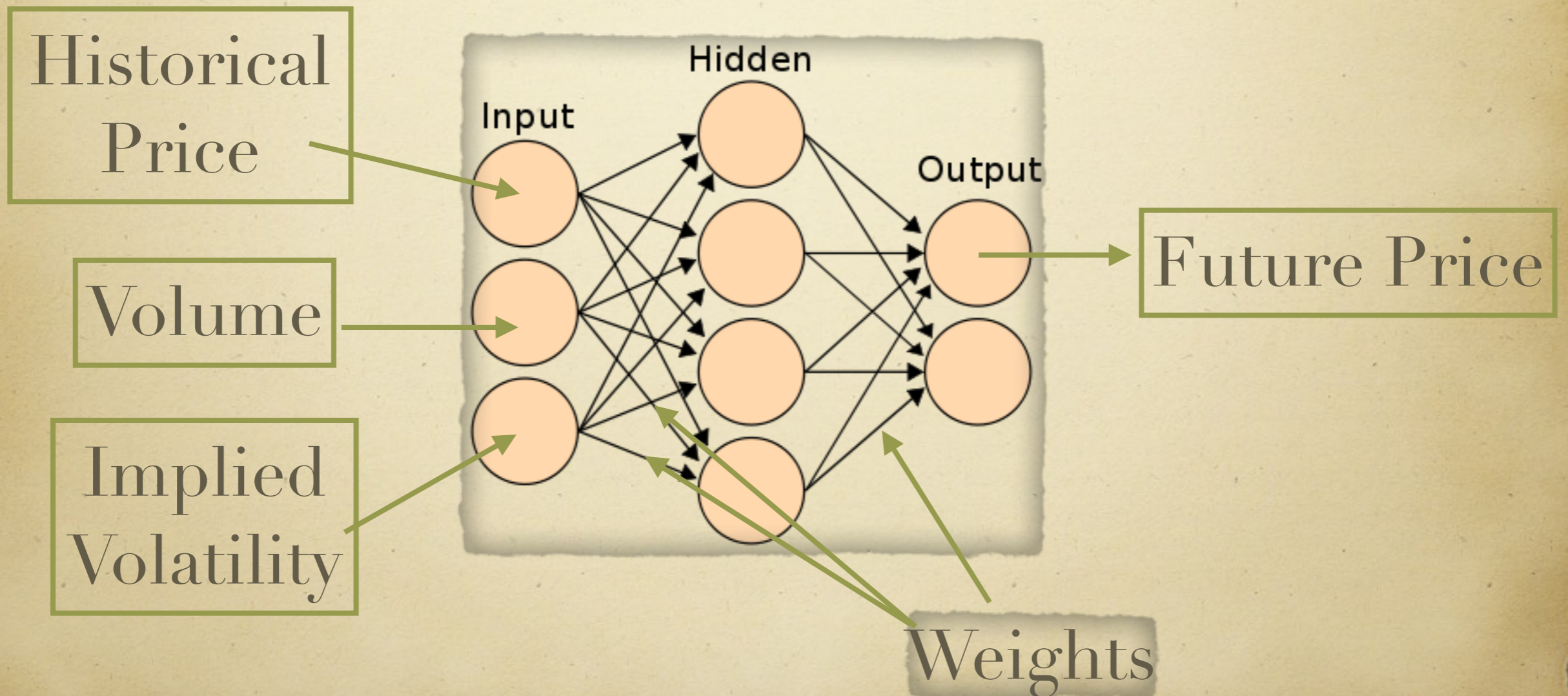
➤ MILLION MILLION MILLION VARIABLES!



A number of ways to predict but neural network

- **Fundamental analysis** - Overall economy and industry's conditions as well as Company's characteristics
- **Technical analysis** - historical price and volume

What is a neural network?

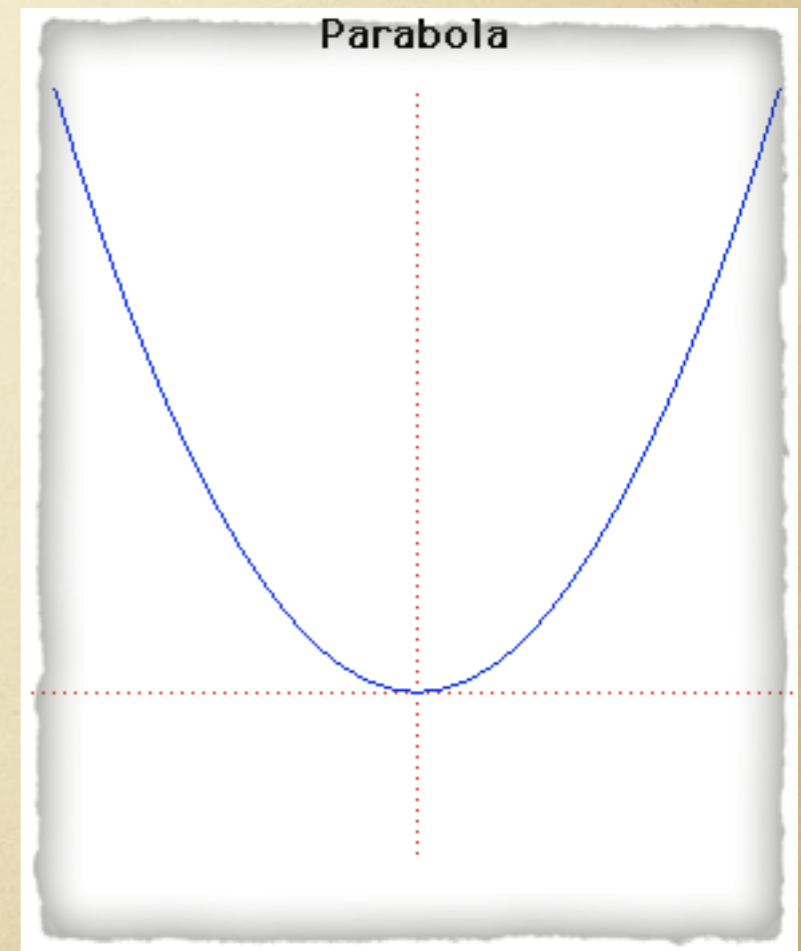


Algorithm

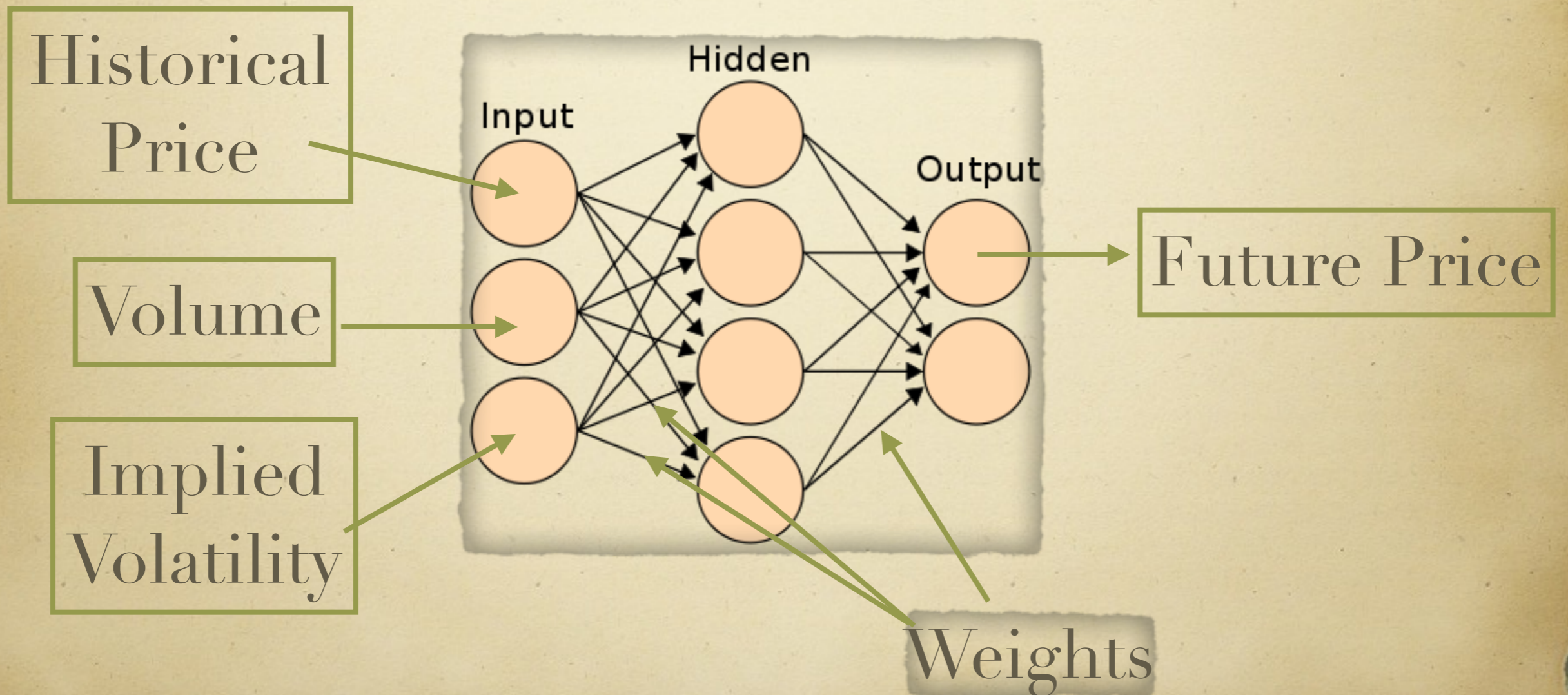
- Cost function = Least squares
- Change Weights of parameters in such a way so that to minimize the cost function.
- Compute Derivative of the Cost Function and successfully minimize it within certain number of steps.

Cost Function

$$J(\theta) = -\frac{1}{m} \left[\sum_{i=1}^m (1 - y^{(i)}) \log(1 - h_{\theta}(x^{(i)})) + y^{(i)} \log h_{\theta}(x^{(i)}) \right]$$
$$= -\frac{1}{m} \left[\sum_{i=1}^m \sum_{j=0}^1 1_{\{y^{(i)} = j\}} \log p(y^{(i)} = j | x^{(i)}; \theta) \right]$$



Back-propagation



Where does the data come from?

- Bloomberg facilities in Sauder
- Internet - Yahoo Finance

Plotting and Visualization

➤ MATLAB

Testing

- Test the neural network on a known function
- Easy to test the actual model because there is A LOT of historical data
- Also, can be tested for future prices

Experimenting with the model

- Change what input variables are
- Change the number of inputs, hidden layers, units in each layer
- Apply to a certain industry and similar market cap companies.

Project Timeline

Oct 20 - 29	Oct 30 - Nov 18	Nov 19 - 21	Nov 22 - Dec 1	Dec 2
Research and Understand the workings of the	Implement code and test on easy functions	Play with parameters, adjust input data	Analyze data and start writing report	Finish report

References

- Machine Learning. coursera.org. Available at: <https://www.coursera.org/course/ml>
- An Introduction to Using Machine Learning to Build Your Trading Strategy. Available at: <https://inovancetech.com/blogML.html>
- CS229 - Machine Learning. Available at: <http://cs229.stanford.edu/>

