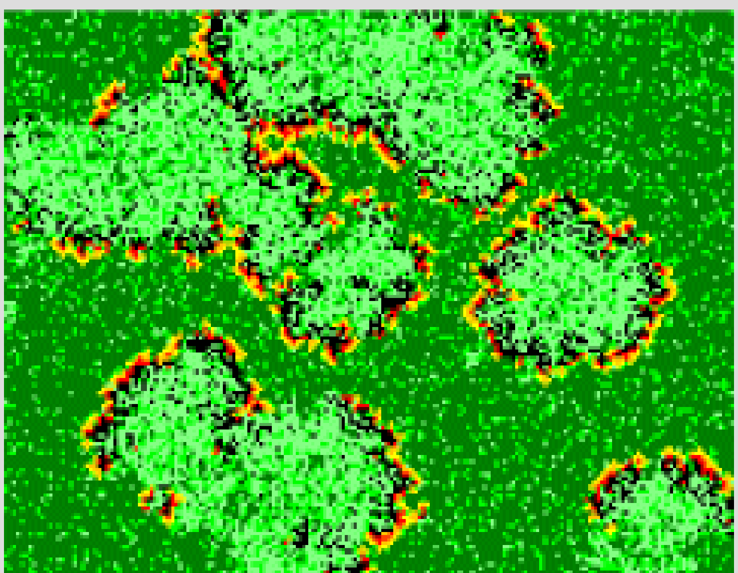
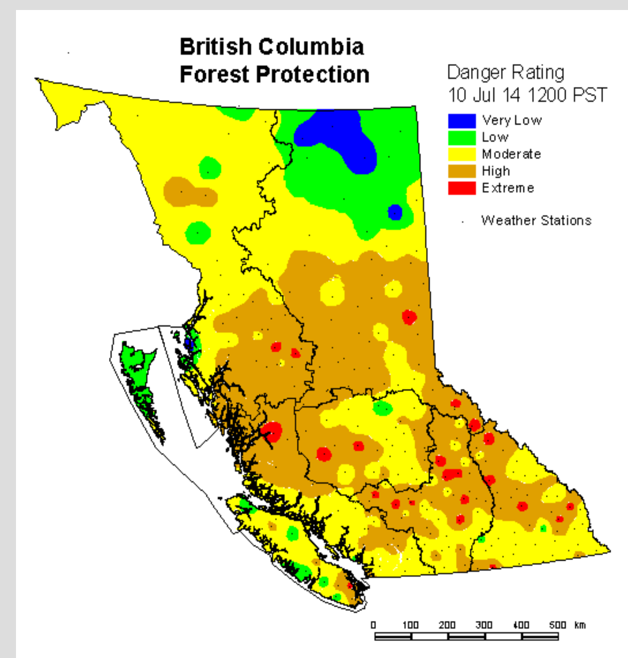
Physics 210 Term Project

Forest Fire Modeling Using Cellular Automata

By: Spencer Austman





Overview

- What is a cellular automaton?
  - A discrete model of a grid of cells that is studied in various scientific domains (including Physics).
  - These cells evolve through a series of time steps according to a set of rules.
- Why do we care about simulating forest fires?
  - The results of the simulations can be used to prevent future fires, protect the ecosystem and to simulate what could happen in a worst case scenario.

Project Goals

- To write a Matlab script that provides an accurate simulation of a forest fire, a graphical output and a statistical output as well.

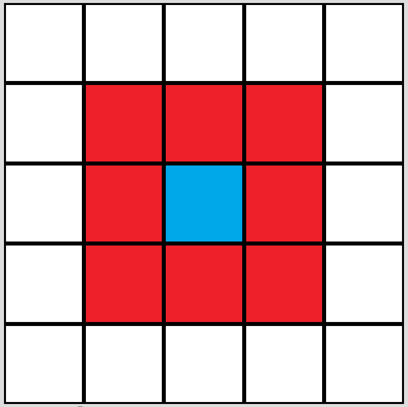
- To examine the effects of certain variables, such as wind and terrain conditions, on the fires and to compare the differences in each scenario.

- To compare my results with other simulations found on the internet.

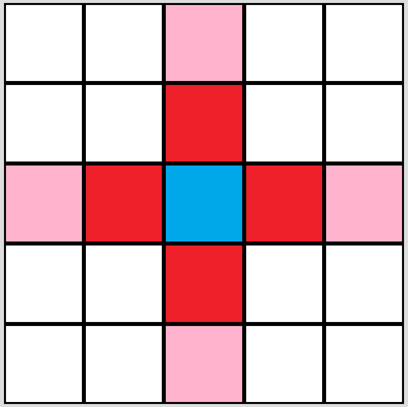
- To simulate extreme cases as to see what would happen in an impossible situation.

Visualization and Plotting Tools

- Cellular automaton are generally represented in one of two ways:
  - Moore neighborhood (which involves a cell and and all eight of its neighbors)



  - Neumann neighborhood (which involves a cell and its four orthogonal neighbors)



Visualization (continued)

- The aforementioned rules:
  - All cells in the cellular automaton are in one of the four states:
    1) Ignitable State (the default state, in other words, an average tree that can be ignited).
    2) Burning State (the cell/tree is on fire)
    3) Burnt State (the fire has extinguished and the tree is no more)
    4) Growing State (this cell can lead to a new tree).
  - Cells are only flammable if the immediate neighbors are in the "Burning State".
  - As in actuality, this cycle will continue on until there are no more trees or no more fire.
  - For simplicity, it is assumed that the terrain flat (allowing a simple 2D model)

Mathematical Formulation

- The simulation is within an n x n lattice where each cell has a coordinate (a,b)
- The transition function for cellular automata:
  - F: $S^n_{(a,b)} \times S^n_{N(a,b)} \rightarrow S^{n+1}_{(a,b)}$
    - This function is dependent on three probability values:
      - P_I : Probability that a cell in the burning state will burn one of its neighbors.
      - P_D: Probability of a cell in the burning state to transition to becoming a cell in the burnt state
      - P_B: Probability of a cell in the ignitable state to transition into a cell in the burning state.
  - S^n represents the state of the cell
    - S^n can represent either the ignitable, burning, burnt or growing state.

Numerical Approach

- Determine how rapidly the fire spread throughout the forest.
- Experiment with various initial conditions, such as the initial location of the fire, and examine the difference in results.
- Examine the effects of variables, such as wind or multiple fires, on the spread of the fire and analyse the different outcomes.

Project Timeline

Dates	Tasks
October 20th → October 24th	Research of the topic, the mathematical formulations needed and the necessary coding.
October 25th → October 30th	Design the code
October 31st → November 9th	Implement the code
November 10th → November 13th	Test the code
November 14th → November 19th	Run the experiment and begin the report
November 20th → November 26th	Analyze the results and finalize the report
November 27th → November 29th	Present the results of the project
November 30th → December 2nd	Submit final draft of the report

References

- http://www.eddaardvark.co.uk/fivecell/forest.gif
- http://assets.vancitybuzz.com/wp-content/uploads/2014/07/forest-fire-risk.gif?7ecf8a
- http://www.sciencedirect.com/science/article/pii/S0307904X06000916

Simulation of a simple neural network

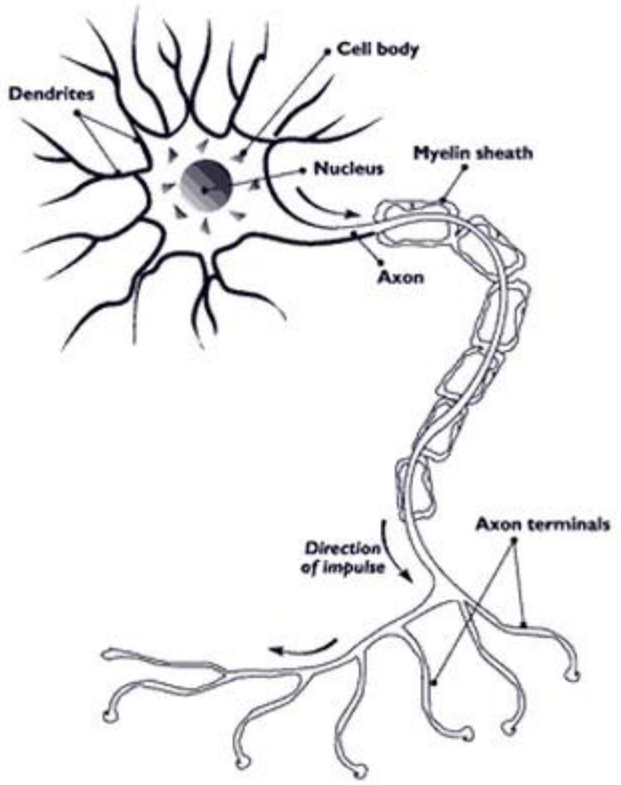
Phys 210 term project proposal

Sophie Boerlage

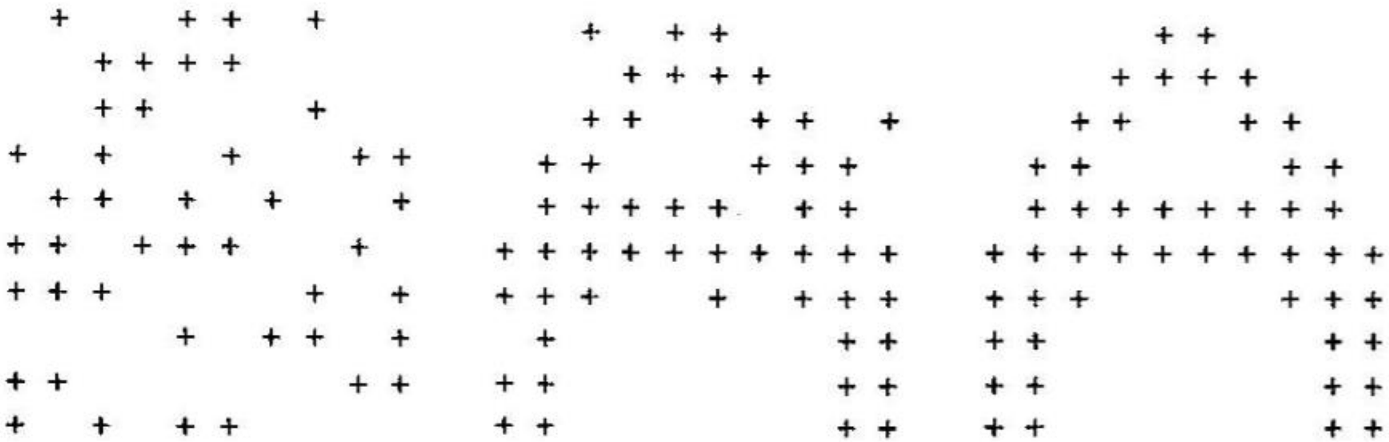
October 21 2014

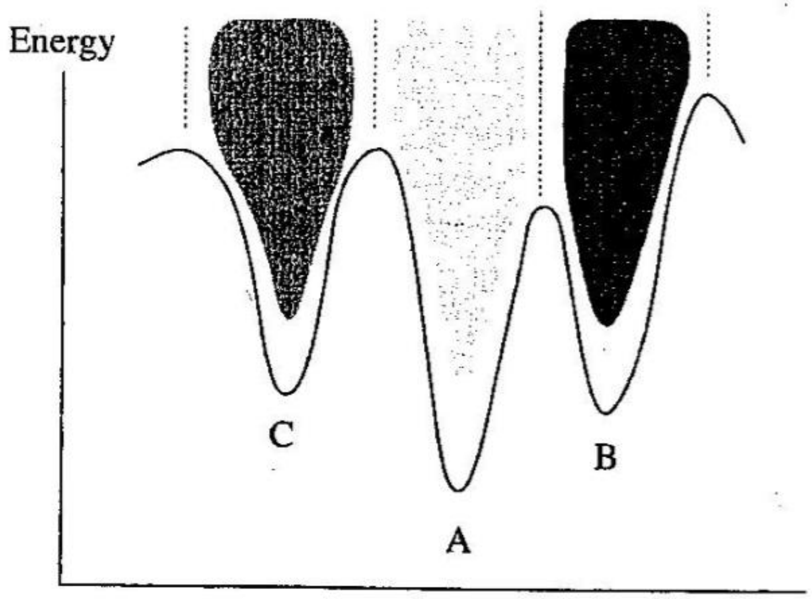
Overview

- A neural network is made up of many interconnected neurons which communicate with each other through electric impulses
- These impulses are triggered by a threshold voltage, and result in the neuron either firing or not firing
- This can be modelled by Ising spin model, in which every neuron is either on or off, or $s = \pm 1$
- Due to the large number of dendrites and axons of each neuron, neurons are able to connect to distant parts of the brain and not just the nearest neighbors



- Pattern will be stored in a lattice of spin values
- Using Monte Carlo method will sweep pattern changing each spin to lowest energy conformation
- Can be used to retrieve stored patterns that differ from the original by varying degrees





Mathematical formulation

Total energy of the system:

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

Where J_{ij} is the connection strength and is equal to:

$$J_{ij} = s_i(m) * s_j(m)$$

And in the case of multiple patterns:

$$J_{ij} = \frac{1}{M} \sum_m s_i(m) * s_j(m)$$

Testing the network

- Model will be tested by:
  - Starting with an increasingly random initial pattern and testing whether the original pattern can be recalled and the number of sweeps needed
  - The experimentally estimated relationship between the number of patterns stored on N neurons is ~ 0.13N before the patterns become unstable and the system stops acting as memory – test this number
  - Test a damaged network where values of J_{ij} are randomly set to zero for the number of sweeps required and whether or not the network can recall the pattern
  - Tech the network new patterns using

$$J_{ij}(new) = \beta J_{ij}(old) + \alpha s_i(p) s_j(p)$$

Where α and β control the speed of learning and forgetting patterns

References

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

Forest Fire Modeling

Luc Briedé-Cooper

Overview

- Forest Fires are naturally occurring events with two major players:
  -trees/plants/brush
  -fire

- Dynamic change over time

- Future states dependent on previous ones

- Consider fire and tree as cells then changing states determined by some *fixed* rule

Project Goals

- Design a Matlab script that models forest fires from a fixed propagation rule
- Create a visual display of the forest fire propagation
- Explore the behavior of forest fires with added variables (to eliminate assumptions where possible)

Formulation/Numerical Approach

- Assumptions:
  - no “wind” or other weather
  - fire cells linger the same amount of time
  - plant re-growth is negligible
  - “flatland” as in flat terrain but also 2D
  - no roads or buildings
- Three kinds of cells:
  -fire
  -forest
  -empty
- The math must adhere to: forest -> fire -> empty
- Initial condition: requires a spark

Testing/Numerical Experiment

- Test multiple sparks (i.e. interacting forest fires)
- Compare simulation to other simulations
- Observe changes in propagation with different time steps

Project Timeline

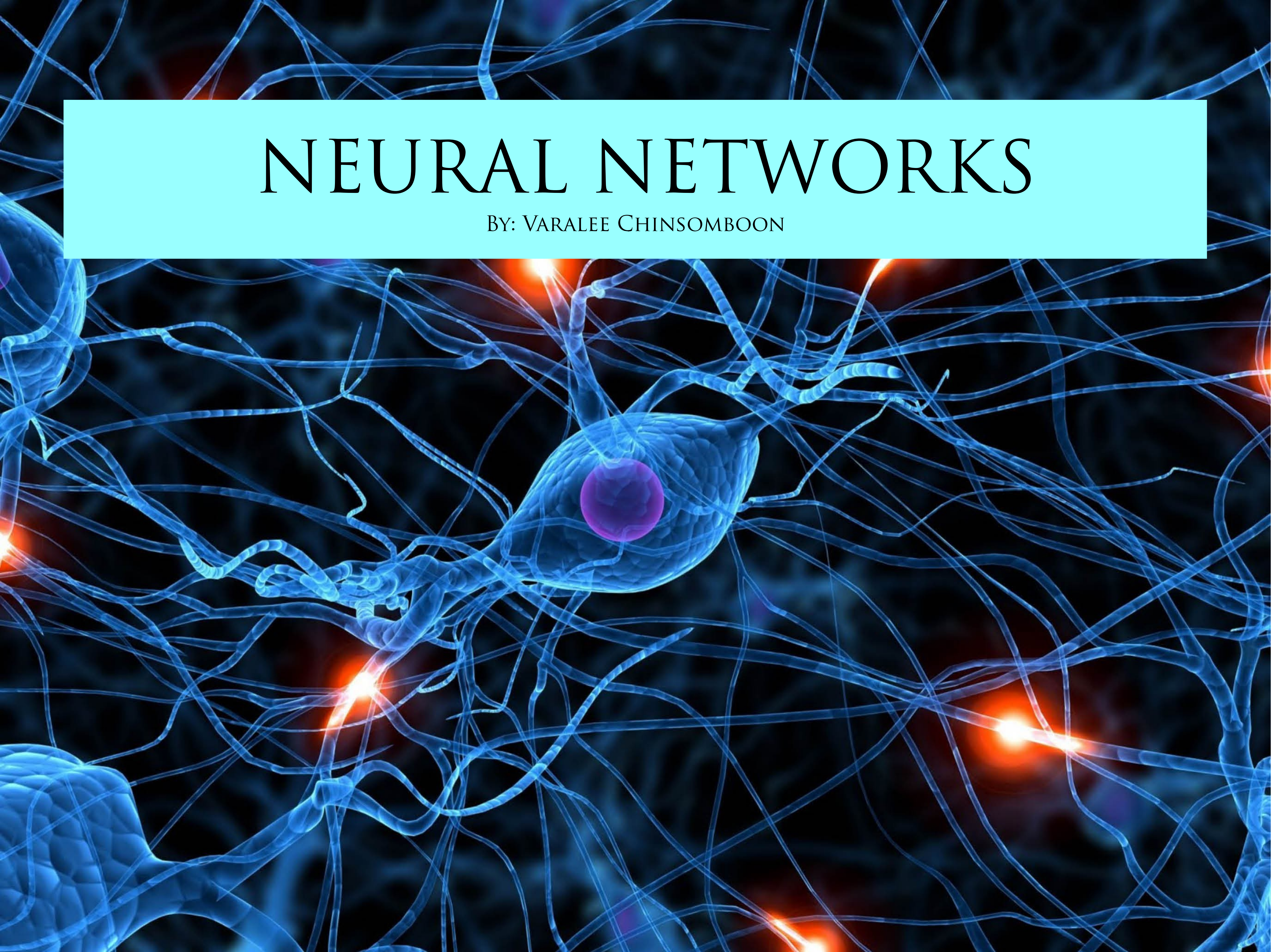
- Oct. 23rd – 28th Basic research
- Oct 28th – November 13th Write Code
- Nov. 13th – 18th Test Code
- Nov. 18th – 25th Analysis and Numerical Experiments
- Nov. 25th – December 3rd Write Report
- Due date: December 3rd!

References

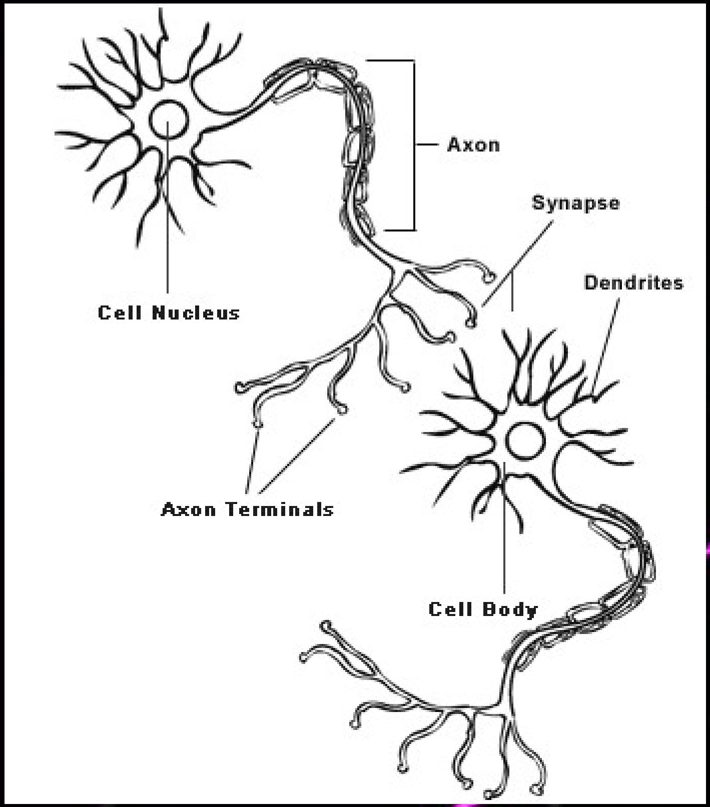
- https://courses.cit.cornell.edu/bionb441/CA/
- http://journals.aps.org/rmp/abstract/10.1103/RevModPhys.55.601

NEURAL NETWORKS

By: Varalee Chinsomboon



Some Background facts



- The main features of a neuron is its axon and dendrites.
- Each neuron is connected to another neuron at a synapse.
- A synapse is where an axon terminal of one neuron meets a dendrite of the other neuron.

- Synapses are either 1) Inhibitory or 2) Excitory
- The sum of inputs that enter through the excitory synapses MINUS the sum of inputs entering through the inhibitory synapses of a neuron is called the FIRING RATE.

Theory

- The firing rate (R) can be expressed as:

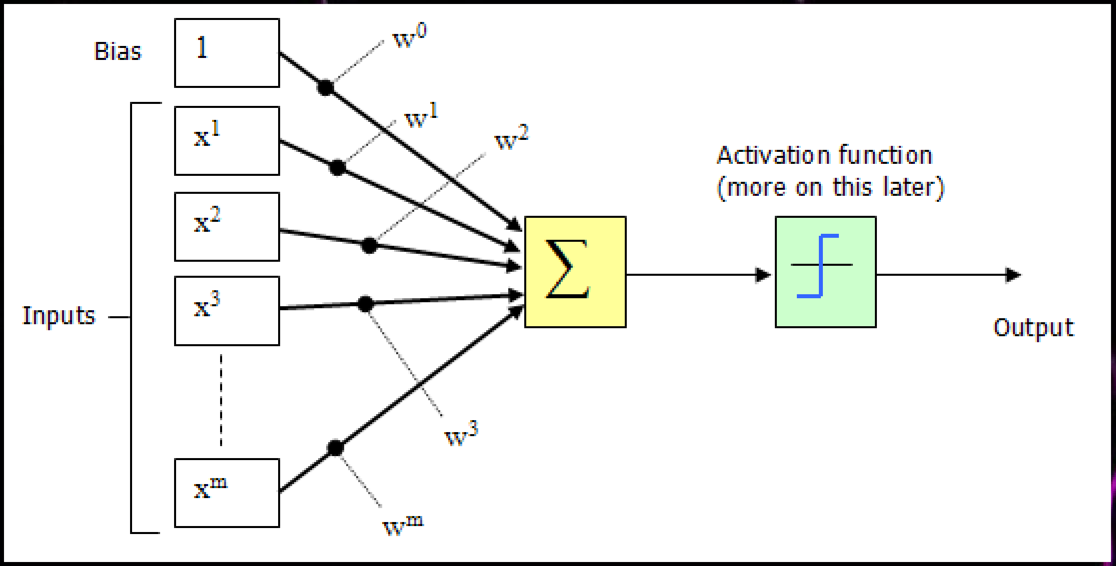
$$R = f(\Sigma V_i)$$

where V_i is the input signal from dendrite 'i' which can be positive or negative

- This firing rate can also be seen as the input of a PERCEPTRON model shown in the corner

- A Perceptron model shows that if the sum of the weighted input passes the set activation function (threshold) then an output is produced.

- For example the activation function could be the sign of the input sum where a positive value will lead to an output and a negative value will not give an output



The Ising model

- The information from the previous slide can be made into the Ising model where the value of the sum of the inputs can be compared to the 2 states of an Ising spin(s_i) such that $s_i = +1$ when the sum is positive or 0 and $s_i = -1$ otherwise.

- An Ising spin can then portray a neuron and so a spin system will be the same as a neural network.

- Our activation function can then be written as the effective energy of the spin system giving:

  $E = -\sum J_{i,j}s_is_j$

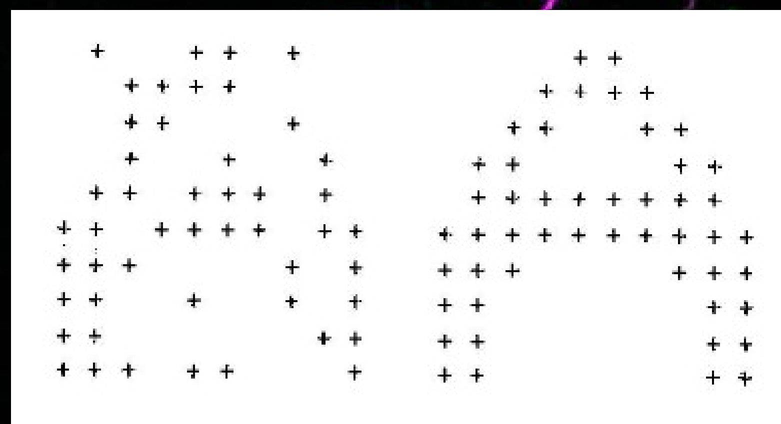
  where J relates to the strength of synaptic connection and $\sum J_{i,j}s_j$ will determine if s_i is positive or negative.

- J for a certain pattern 'm' and the total number of patterns 'M' can be written as : $J_{i,j} = 1/M(\sum J_{i,j}s_i(m)s_j(m))$

- Giving the overall equation: $E = -\sum 1/M(\sum J_{i,j}s_i(m)s_j(m))s_is_j$

Pattern recognition

- We can use the Ising model with the Monte Carlo algorithm to match a contorted pattern with its ideal pattern, for example:



- This is done by the algorithm comparing the spin value with the effective energy E. Where the spin value will be flipped if the flipped value is more negative, otherwise the spin will be left alone.
- This is repeated over time until none of the spins are in need of flipping (stable state) and the resulting pattern is the ideal pattern.

Project timeline

Dates	Activities
10/19-10/26	Basic research, Begin code design
10/27-11/16	Implement code
11/17-11/19	Test code
11/20-11/25	Run numerical experiments, analyze data, begin report
11/26-11/28	Finish and submit report?

References

- Computational Physics (ch.12.4) by Nicholas J. Giordano
- The nature of code (Daniel Shiffman) http://natureofcode.com/book/chapter-10-neural-networks/
- Past students' presentations
- Matt's presentation
- http://www.codeproject.com/Articles/16419/AI-Neural-Network-for-beginners-Part-of

Toomre Model of Galaxy Collisions

Phys 210 Project Proposal

Zaeema Choudhri

Overview

- In the 1970s, the Toomre brothers conducted the first computer simulations of galaxy collisions
- The Toomre model simplifies the simulations of the collisions by making the following assumptions:
  - Negligible gravitational attraction between individual stars
  - Negligible masses of the stars; will only consider galactic nuceli, which contains most of the mass
  - No dark matter is present
  - No external forces

Goals

- To compose a MATLAB code that simulates galaxy collisions in accordance with the Toomre model
- To analyze how changing the initial variables affects the collision, i.e. initial velocities, number of stars, core masses
- To test the validity of the model by comparing it to actual galaxy collisions
- To measure the accuracy of the model in predicting said collisions

Mathematical Formulation

- Newton's Law of Gravitation: $F = \frac{GMm}{r^2}$
- Newton's Law of Motion: $F = ma = \frac{mv^2}{r}$
- Kepler's Third Law: $p^2 = \frac{4pi^2}{G(M+m)}a^3 \approx \frac{4pi^2}{GM}a^3$

Numerical Approach

- Will consider two scenarios: 1) one galaxy has an initial velocity while the other is motionless and 2) both galaxies have initial velocities

- Since the gravitational forces will affect the accelerations of the galaxies, I will use the finite difference approach to calculate said accelerations
  - the accelerations will produce new velocities, which will also be computed using FDA
  - note: I will only consider the gravitational forces between the stars and galactic nuclei, i.e. attraction between the stars will be ignored as the masses of the stars are relatively small

Testing and Numerical Experiments

- Vary the initial parameters including, but not limited to, angle of approach, the size (e.g. mass, star count) of the galaxies, initial velocities and initial positions
- Compare results with the Toomre model of the collision of the Antennae galaxies
- Compare results with actual (observed collisions)

Project Timeline

Dates	Activities
10/18 – 10/28	Conduct basic research, find all relevant equations, begin coding
10/28 – 11/18	Implement code
11/18 – 11/22	Test code
11/22 – 11/28	Carry out numerical experiments, analyze the data, begin report
11/28 – 12/02	Complete report
12/02	Submit project

References

- http://en.wikipedia.org/wiki/Alar_Toomre
- http://en.wikipedia.org/wiki/Antennae_Galaxies
- http://en.wikipedia.org/wiki/Interacting_galaxy
- http://cas.sdss.org/dr6/en/proj/basic/galaxies/collisions.asp

Questions/comments?

The Alar Toomre Model of Galaxy Collisions

Phys 210

Ben Chugg

October 21, 2014

What is the Toomre Model?

- Models of intergalactic collisions are extremely complex
- Toomre developed a particle model that made certain simplifying assumptions which was able to explain certain observed phenomena (Distorted tidal tails found in antennae galaxies)
- ISM (Interstellar Medium) and Dark Matter are ignored
- Mass of system assumed to lie almost entirely at the galactic core

Project Goals

- To write a Matlab/Octave code which will allow me to simulate galaxy collisions in 3 – dimensions
- If my capabilities allow, to model the collision of binary galactic system with a larger spiral galaxy (3 galaxies total)
- To implement the code under different initial conditions (ie. Velocities, orientations and positions)

The Maths – Key Equations

$$\mathbf{F} = \frac{\mathrm{d}\mathbf{p}}{\mathrm{d}t} = \frac{\mathrm{d}(m\mathbf{v})}{\mathrm{d}t}.$$

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

➔ $$\ddot{\mathbf{r}}_\mathbf{i} = G\sum_{j\neq i} m_j r_{ij}^{-2}\hat{\mathbf{r}}_{ij}$$

(Above) : Newton's classical mechanics equations (Where n dots denotes the nth derivative)

(**Boldface** represents vector notation)

$$\frac{T^2}{r^3} = \frac{4\pi^2}{GM} \quad \text{(for } M>>m\text{)}$$

(Above): Kepler's Law of Planetary Motion (Applied to stars and and galactic center of mass)

$$\mathbf{R}_0 = \mathbf{R}_f + \mathbf{v}\Delta t$$

➔ $$< r_{x0}, r_{y0}, r_{z0} > = < r_{xf}, r_{yf}, r_{zf} > + < v_x, v_y, v_z > \Delta t$$

$$-\frac{GmM}{|\mathbf{r}|^2}\hat{\mathbf{e}}_r + \mathbf{R} = m\frac{d^2\mathbf{r}}{dt^2} + 0 \quad \Rightarrow \quad \frac{d^2\mathbf{r}}{dt^2} = -\frac{GM}{|\mathbf{r}|^2}\hat{\mathbf{e}}_r + \mathbf{A}$$

More Maths!

Things tend to get a little more complicated in 3-dimensions. The general equations of motion that will be governing my particles (using spherical coordinates) – where boldface once again represents vector notation - are :

$$\mathbf{r} = \mathbf{r}(t) = r\hat{\mathbf{e}}_r$$

$$\mathbf{v} = v\hat{\mathbf{e}}_r + r\frac{d\theta}{dt}\hat{\mathbf{e}}_\theta + r\frac{d\phi}{dt}\sin\theta\hat{\mathbf{e}}_\phi$$

$$\begin{aligned}\mathbf{a} = &\left(a - r\left(\frac{d\theta}{dt}\right)^2 - r\left(\frac{d\phi}{dt}\right)^2\sin^2\theta\right)\hat{\mathbf{e}}_r \\ &+ \left(r\frac{d^2\theta}{dt^2} + 2v\frac{d\theta}{dt} - r\left(\frac{d\phi}{dt}\right)^2\sin\theta\cos\theta\right)\hat{\mathbf{e}}_\theta \\ &+ \left(r\frac{d^2\phi}{dt^2}\sin\theta + 2v\frac{d\phi}{dt}\sin\theta + 2r\frac{d\theta}{dt}\frac{d\phi}{dt}\cos\theta\right)\hat{\mathbf{e}}_\phi\end{aligned}$$

The coordinates (r, theta and phi) correspond to the respective unit vectors $\mathbf{e}_r$,$\mathbf{e}_{theta}$,$\mathbf{e}_{phi}$

Numerical Approach

- Thanks to Newton, we can represent the force on a object in terms of its first and second derivatives. These derivatives can be approximated using first and second order FDA's
- These Finite Difference Approximations will model how the particles are moving over a discrete grid with respect to x,y,z and t.

Timeline

Dates	Ideal Progress
October 22 - 30	Further Research, derive equations and begin code design
October 31- November 10	Write script
November 10 - 18	Test script, run experiments, begin report
November 25 - 28	Finish Report
November 28	Submit Project

References

- East Tennessee University, Department of Physicshttp://faculty.etsu.edu/smithbj/collisions/collisions.html
- UBC Physics and Astronomy, Numerical Relativity http://laplace.physics.ubc.ca/210/Proposals-2012/L1A.pdf
- http://en.wikipedia.org/wiki/Equations_of_motion


Questions?
Comments?
… Complaints?

GRAVITATIONAL N-BODY PROBLEM

PHYS 210 Term Project Proposal

Kyle de Jong

OVERVIEW

- The gravitational n-body problem is a model of the interaction of objects in space via forces
- These forces are gravitational, and cause an acceleration on all objects involved
- These objects all orbit around a point known as the centre of mass

PROJECT GOALS

- To write code in MATLAB that models the n-body problem, including forces, acceleration, velocity, and positions of all objects involved
- Create a simulation that shows the motion of the objects in two dimensions

MATHEMATICAL FORMULATION

- The fundamental equation for this topic will be Newton's law of universal gravitation:

$$\overrightarrow{F_{i,j}}(t) = G\frac{m_i m_j \overrightarrow{\Delta x_{ji}}(t)}{\left|\overrightarrow{\Delta x_{ji}}(t)\right|^3} = m_i \frac{d^2 x_i(t)}{dt} = m_j \frac{d^2 x_j(t)}{dt}$$

- The accelerations of the objects can be found by the finite difference approximation of the second time derivative:

$$a(t) \equiv \frac{d^2 x(t)}{dt} = f''(x_j) \equiv \frac{f_{j+1} - 2f_j + f_{j-1}}{\Delta x^2}$$

MATHEMATICAL FORMULATION (CONT.)

- The velocity is defined as the first time derivative of position:

$$v(t) \equiv \frac{dx(t)}{dt} = f'(x_j) \equiv \frac{f_{j+1} - f_{j-1}}{2\Delta x}$$

- The objects orbit around a centre of mass defined as:

$$x_{\text{cm}} = \sum_{i=1}^{N} m_i x_i \,/M$$

NUMERICAL APPROACH

- With initial position and velocity chosen, the force and therefore the accelerations of the objects can be determined
- Actual computation techniques will be covered in future lectures/labs, and so the numerical approach is not fully known at this point and time

VISUALIZATION AND PLOTTING TOOLS

- MATLAB will most likely be used for plotting
- Programs introduced during upcoming lectures/labs may be used as well

ASSUMPTIONS

- The objects are perfectly spherically symmetrical
- The only forces acting on the objects are that of gravitation from the other massive objects; no external forces are involved
- The orbits are circular, with eccentricity being non-existent

TESTING AND NUMERICAL EXPERIMENTS

Testing

- Ensure behaviour of data reflects that of an actual planetary orbit

- Use fixed time intervals through a process of discretization

Numerical Experiments

- Using Newton's Law of universal gravitation, see if in fact the formula holds true with N arbitrary masses via simulation of a system using the finite difference approximation with first and second order differential equations

PROJECT TIMELINE

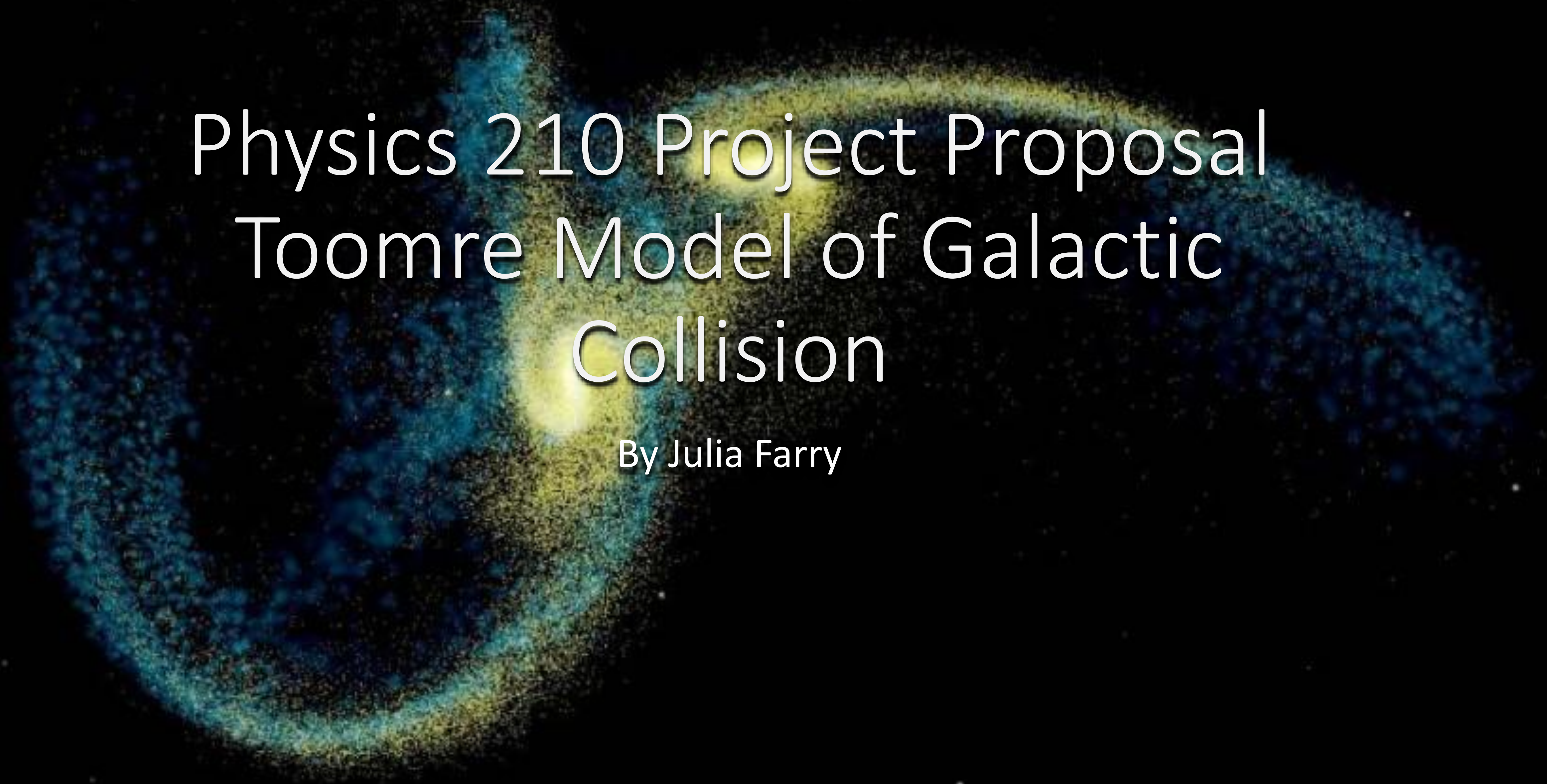
Dates	Activities
10/19 – 10/30	Do research on the topic; find all relevant equations
10/31 – 11/18	Begin coding
11/19 – 11/23	Test authenticity of coding
11/24 – 11/30	Run multiple experiments with the code, make sure everything is running properly
11/31 – 12/3	Start and complete write-up report
12/4	Final revision of the project and submission of report

REFERENCES

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

Physics 210 Project Proposal Toomre Model of Galactic Collision

By Julia Farry



The Project

What's the Toomre Model all about?

In the early 70's Alar Toomre modelled a galactic collision for the first time with computers. By reducing the number of stars to 1000 he proposed ways in which disk shaped galaxies would interact.

What's the plan?

- Base the parameters off Toomre Model
- Simulate a collision of galaxies with higher N-body count.

The Mathematical Basis

- From Classical Mechanics we must consider following equations:
  - Gravitational Force: $F = \frac{GMm}{r^2}$
  - Centripetal Force: $F = \frac{mv^2}{r}$
- To apply these equations the masses of the galaxies are considered by the sum of N stars, i.e. the center of mass of the galaxy
- Assumptions:
  - The galaxies are disk shaped
  - The galaxies move on elliptical or parabolic paths.

Ellipse: $\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$ for a>b

Parabola: $y = ax^2 + bx + c$

Numerical Approach and Experiments

In order to appropriately parameterize the simulation, we must consider the magnitude of the interactions:

- Mass: Order of 10^{12} solar masses or 2×10^{42}kg
- Velocity: 300 kms^{-1}
- Time scale: 3×10^{8} years

Experimentation will be done by varying initial conditions by mass, velocity and energy (i.e. energy from spin and motion). The difference in mass between the galaxies will be considered as well.

- Visualization and Plotting Tools;
  - Since this is a simulation of galaxy collisions generation of animations will be required. MATLAB will be used for any mathematical analysis.

- Testing
  - Check the result of the simulation so see if it shows any signs of galactic bridges and tails.
  - Compare results with Toomre and Toomre modelling.

Timeline for Project

Dates	
10/16 – 10/24	Basic research, make basis of equations and start design for code
10/25 – 11/15	Implement code, get to the bugs early
11/16 – 11/19	Test code
11/20 – 11/25	Run numerical experiments, analyze data, and begin writing the report.
11/26 – 12/03	Corrections and proofread; Submit as soon as its ready.

References

http://www.public.iastate.edu/~curt/cg/section1.html
http://www.cv.nrao.edu/~jhibbard/students/CPower/numerical/num_antennae/tt72.html
http://www-physics.ucsd.edu/students/courses/winter2010/physics141/final/final.html

SIMULATIONS ON OPTICS:

MODELLING THE RAY TRACING USING MIRROR AND LENS

Merrill Cheuk Kiu Fung
11654143

OVERVIEW

- The idea of the project is to model light as a particle that moves at constant speed and in a fixed direction
- It will changes direction until it encounters some optics devices. For example, mirror and lens

OVERVIEW

- (1) Mirror are the basic device in optics which reflects light.

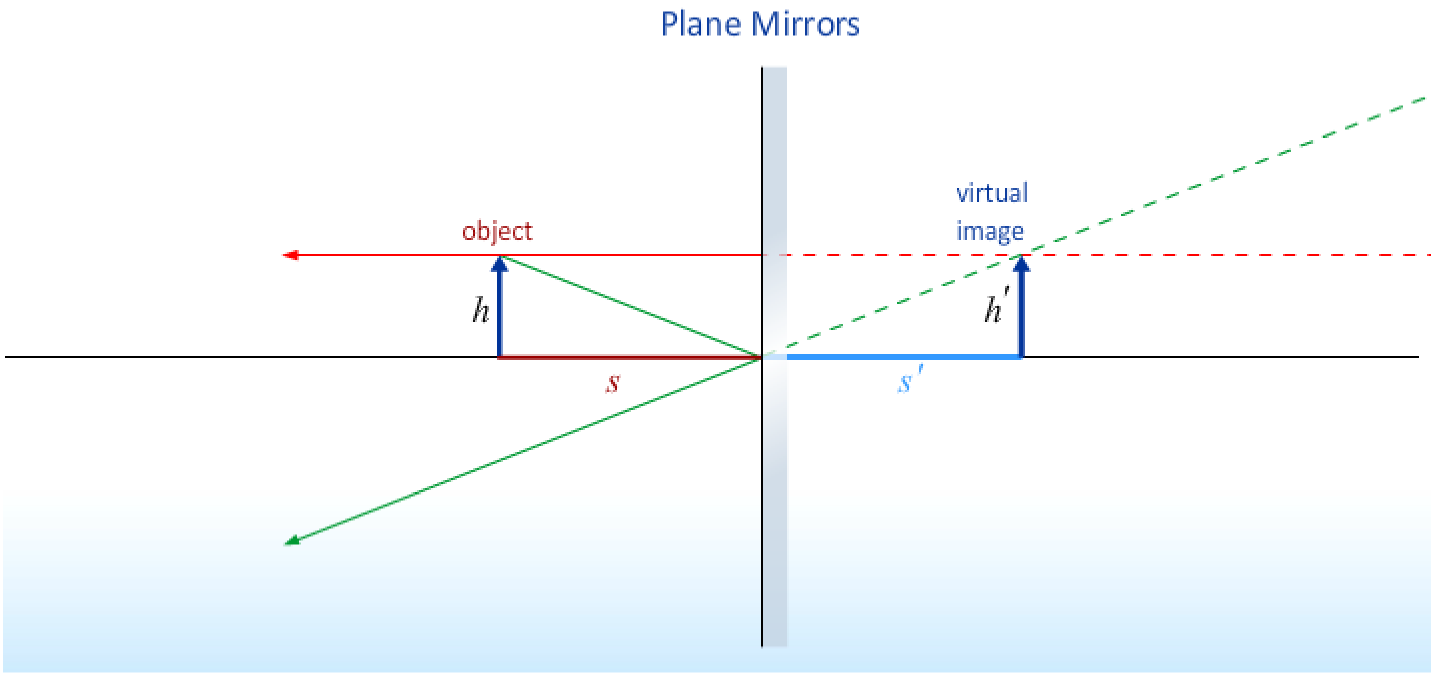
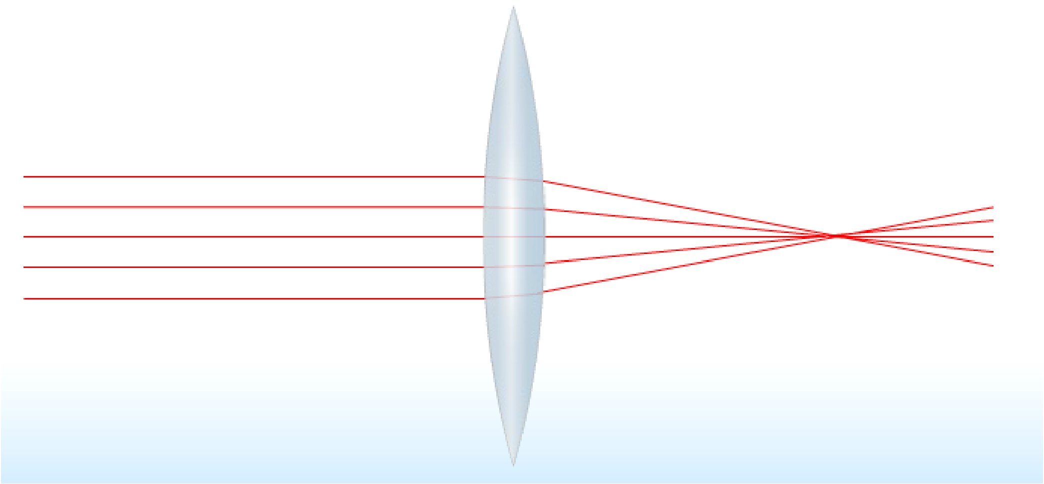


Image credits: https://www.smartphysics.com/Course/PlaySlideshow?unitItemID=192062&enrollmentID=28961

OVERVIEW

- (2) Lens are the basic device in optics which transmits and refracts light (Change in refractive index)

Converging lens



Diverging lens

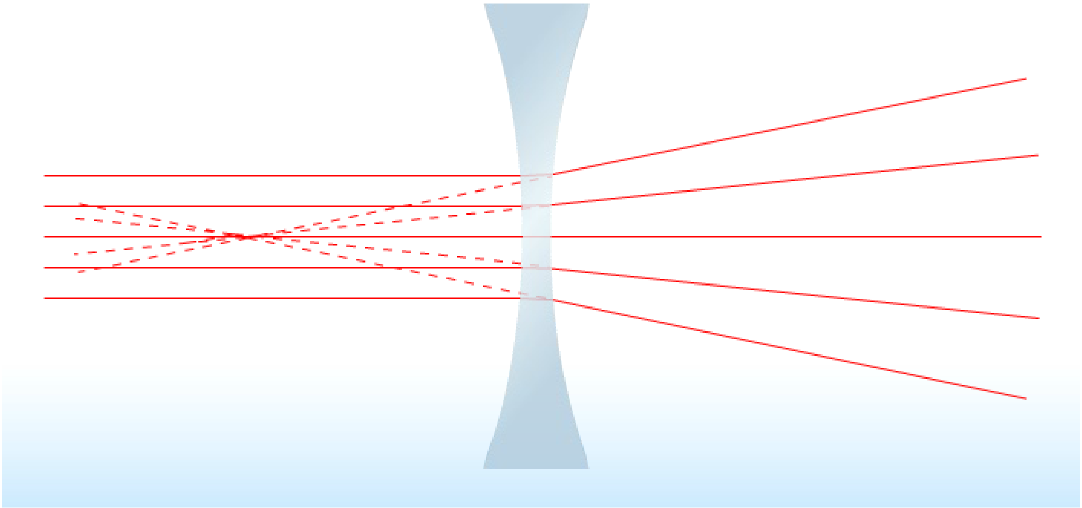


Image credits: https://www.smartphysics.com/Course/PlaySlideshow?unitItemID=192062&enrollmentID=28961

PROJECT GOALS

- To write an MATLAB code which simulates the ray tracing of both of the mirror and lens
- The model is implemented as a simulation where in the motion of a single particle is computed based on a set of rules
- Knowledge Deficit: Similar work has been done by Alex Fang. Yet, I will try using the simulation to explore the relationship between Object and image distance

Image credits: https://www.smartphysics.com/Course/PlaySlideshow?unitItemID=192062&enrollmentID=28961

MATHEMATICAL FORMULATION (EQUATIONS OF LENS)

- The motion of particle when encounter the mirror will follow by the law of reflection

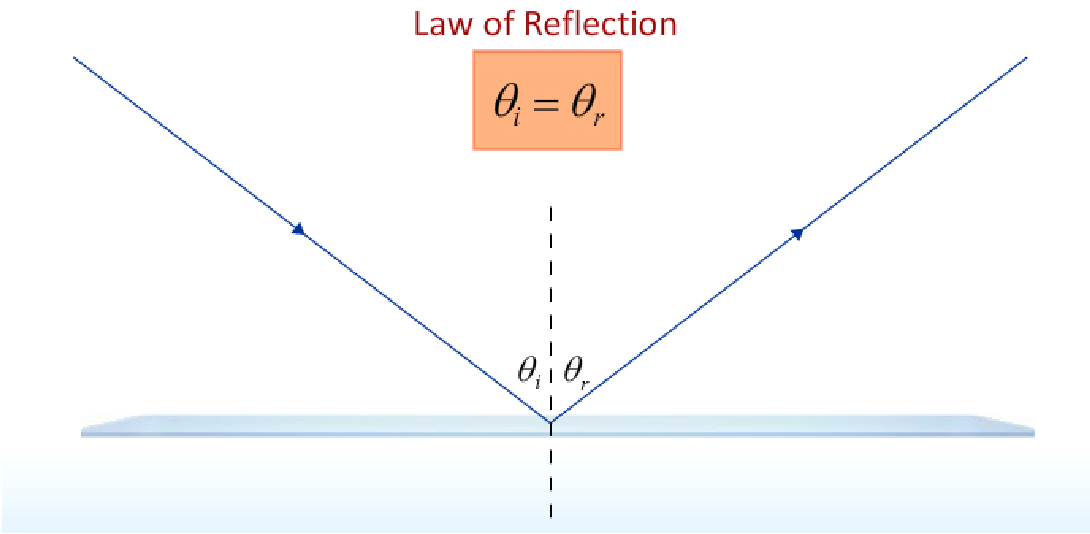


Image credits: https://www.smartphysics.com/Course/PlaySlideshow?unitItemID=192062&enrollmentID=28961

MATHEMATICAL FORMULATION (EQUATIONS OF LENS)

- The motion of particle when encounter the lens will follow by the Snell's Law

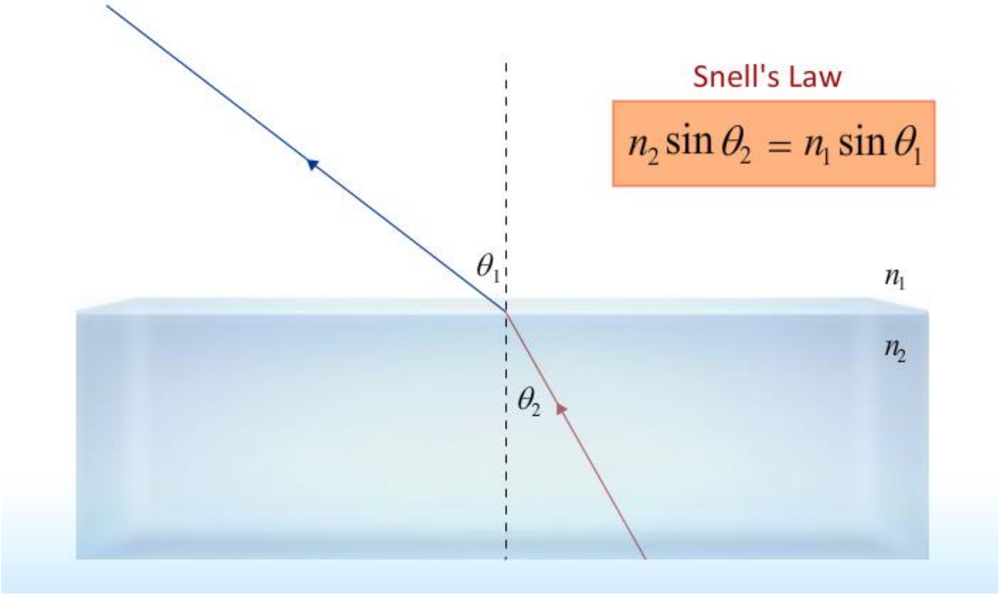


Image credits: https://www.smartphysics.com/Course/PlaySlideshow?unitItemID=192062&enrollmentID=28961

TESTING

- Successful model will be tested with an arbitrary set of mirrors.
- Think and thin lens would also be tested

Image credits: https://www.smartphysics.com/Course/PlaySlideshow?unitItemID=192062&enrollmentID=28961

PROJECT TIMELINE

Dates	Goals
10/13–10/24	Do basic research, derive equations
10/25–11/15	Do basic coding
11/16–11/19	Test code
11/20–11/25	Run simulation
11/26-11/28	Finish report

Image credits: https://www.smartphysics.com/Course/PlaySlideshow?unitItemID=192062&enrollmentID=28961

REFERENCE

- https://www.khanacademy.org/test-prep/mcat/physical-processes/thin-lenses/v/object-image-and-focal-distance-relationship-proof-of-formula
- https://www.smartphysics.com/Course/PlaySlideshow?unitItemID=192062&enrollmentID=28961

Creating Traffic Simulations Using Cellular Automata

PHYS 210 Term Project Proposal

Cattleya Grant

Overview

- Traffic patterns can be modelled using an array containing cells which are in one of two states: empty or filled
- Each filled cell represents a car, and each car will be affected by factors such as the behaviour of the cars around it, and its own speed and acceleration.

Project Goals

- To write a code in Matlab to model traffic patterns using probability functions
- To check the correctness of the code through comparison of the results with past studies
- To look at both closed loop and open traffic systems to analyze the conditions at which traffic jams occur

Mathematical Formulation

- Equation for the probability for each cell of it being occupied at time t:

$$\frac{dP(\sigma_i)}{dt} = -\sum_i W(-\sigma_i, -\sigma_{i+1}|\sigma_i, \sigma_{i+1})P(\sigma_i, \sigma_{i+1}, t) + \sum_i W(\sigma_i, \sigma_{i+1}| -$$
$$\sigma_i, -\sigma_{i+1})P(\sigma_i, \sigma_{i+1}, t)$$

  - Each sigma has a value of +/- 1, with 1 for occupied and -1 for empty cells
- Other useful formulas:
  - $$density = \rho = \frac{1}{T}\sum_{t=t_o+1}^{t_o+T} n_i(t)$$

Numerical Approach

- With each update in t, there are four components of a filled cell to consider:
  - Acceleration (e.g. a car not at maximum velocity and behind a car travelling at a higher v will accelerate)
  - Slowing down (e.g a car will slow down if it is behind a car where the distance to the car in front (j) is lower than the car speed (v))
  - Randomization: (a probability p will be used to cause randomly chosen cars to decrease (v) by 1)
  - Car motion (at every t, a car will advance (v) cells)

  This yields the following variables: the velocity (v), the position of the car (i), the position of the car in front (j), the time (t), and the randomization probability (p)

Testing and Numerical Experiments

- Compare both a closed loop and an open loop system where the total number of cars and the density may change
- Use a variety of initial conditions such as different densities or different average velocities of the vehicles

Project Timeline

Oct. 22/14- Oct25/14	Research, basic code design
Oct 25/14- Nov.8/14	Implement and test code
Nov.8/14- Nov.22/14	Run experiments and analyze results
Nov.22/14-Nov.29/14	Write report
Nov.30/14	Final checks on report, submit report

References

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

- K. Nagel and M. Schreckenberg, “A cellular automaton model for freeway traffic”. *J. Phys. I* (1992): 2221-2229. Web.

Finite Differencing Approach to the Thompson Problem: Charges on a Sphere

PHYS 210 Term Project Proposal

Chan Gwak

21 October 2014
Tuesday

Charges on a Sphere

- Overview
  - The Thomson problem deals with finding the minimum electrostatic potential energy configuration of *N* identical charges on a sphere.
  - These charges repel each other with a force dependent on the distance between them and should approach the minimum energy state on their own if given enough time (i.e. progress to equilibrium), the state intuitively predicted to be the "most spread out" state.
- Goals
  - To write a code on MATLAB to describe the motion of the charges on the surface of the sphere with second-order finite difference approximations
  - To test and correct the code based on known results
  - To use the model to try and determine configurations for values of *N* for which the arrangement is not well understood.

Mathematical Formulation

- The force that one electric charge applies on another is described by the equation:

$$\vec{F} = m\vec{a} = \frac{kq_1q_2}{|\Delta r|^2}\widehat{\Delta r} = kq_1q_2\left(\frac{\Delta r}{|\Delta r|^3}\right)$$

- The Coulomb constant k, the charges of the points q and the masses of the points m will be simplified to be 1. Thus, we get the components of each acceleration vector:

$$\frac{d^2x}{dt^2} = \frac{\Delta x}{\left((\Delta x)^2 + (\Delta y)^2 + (\Delta z)^2\right)^{\frac{3}{2}}}$$

$$\frac{d^2y}{dt^2} = \frac{\Delta y}{\left((\Delta x)^2 + (\Delta y)^2 + (\Delta z)^2\right)^{\frac{3}{2}}}$$

$$\frac{d^2z}{dt^2} = \frac{\Delta z}{\left((\Delta x)^2 + (\Delta y)^2 + (\Delta z)^2\right)^{\frac{3}{2}}}$$

- As there are many charges acting on each other, the force on each charge will be the sum of the forces exerted on it by the other charges.
- Furthermore, the charges will be constrained to the surface of a sphere of radius 1, so for each point, the coordinates must obey

$$r^2 = x^2 + y^2 + z^2 = 1$$

(The direction of motion will be tangent to the sphere.)

- The simulation will be run on the domain

$$-1 \leq x \leq 1$$
$$-1 \leq y \leq 1$$
$$-1 \leq z \leq 1$$
$$0 \leq t \leq t_{max}$$

with the points beginning from randomly scattered positions in a defined section of the sphere.

Numerical Approach

- As a finite difference approximation, the domain will be replaced with a discrete lattice of points.
- The equations of motion involved will be approximated using second-order FDAs.

Visualization and Plotting Tools

- TBD

Testing

- Run simulations each with half the discretization scale of the previous run and ensure that the error term converges as expected
- Confirm that the model works for values of N for which the configurations are known (e.g. $N=4$ results in a tetrahedral arrangement)

Numerical Experiments

- Find values of N at which new discernible configurations seem to appear and investigate these.

Project Timeline

Date	Tasks
13 Oct – 28 Oct	Research & Finalize Equations
28 Oct – 20 Nov	Implement and Test Code
21 Nov – 24 Nov	Experiments & Data Analysis, Begin Report
25 Nov – 2 Dec	Complete Report
2 Dec	Submit Project

References

- "Thompson problem." Wikipedia. http://en.wikipedia.org/wiki/Thomson_problem. Retrieved 19 Oct, 2014.

PHYS 210 Term Project Proposal

N-body Problem

Zhicheng Jiang

Overview

- **Interaction of particels due to gravitational force.**
- **Predicting the motions of *n* objects under mutual force**.

Project Goals

- To write a MATLAB (octave) code that describes the motion of N particles due to gravitational forces.
- Create a simulation of these interactions.
- Test the results.

Relevant Equations

- **Gravitational force:**

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

m1 & m2 = mass of objects.
r = distance between them.
G = gravitaional constant = 6.67300 x 10^{-11} $m^3 kg^{-1} s^{-2}$.

- **Newton's second law F=ma:**

$$\frac{d^2\vec{x}}{dt^2} = \vec{a} = \frac{Gm_2}{r^2}\hat{r}$$

Numerical Analysis

- Assuming acceleration and velocity being are constant over Δ t.
- Using finite difference approximations (FDA) to evaluate how force changes the acceleration and velocity of each objects.

Testing and Numerical Experiment

- Calculate total energy and momentum to make sure they are conserved.
- Investigate the system by using different number of objects.
- Investigate the system by changing the initial values for the objects (mass, positon and velocity).

Timeline

Dates	Activities
10/20 - 10/26	Research & design code
10/27 - 11/15	Implement Code
11/16 - 11/20	Test code
11/21 - 11/26	Run numerical experiments, analyze data, begin report
11/26 - 12/1	Finish report
12/1	Submit project

References

- http://en.wikipedia.org/wiki/N-body_problem
- http://en.wikipedia.org/wiki/Newton's_law_of_universal_gravitation
- https://www.princeton.edu/~achaney/tmve/wiki100k/docs/N-body_problem.html

Freeway Traffic Model Utilizing Cellular Automata

PHYS 210 Term Project Proposal

October 2014

Brian Kim

Overview

- In freeway traffic it can be seen that once vehicle density increases to a certain point traffic stops flowing freely and start-stop waves begin to propagate

- This phase change between laminar flow to turbulent behaviour is an interesting and unique property of traffic

Project Goals

- Construct a working model of one-dimensional flow of freeway traffic using cellular automata methods and MATLAB

- Run and analyze the model and compare results to data collected from real traffic as well as model data from another study

- Apply open and periodic boundary conditions and different initial conditions (traffic densities, speeds)

Mathematical Formulation

The model is defined on a one dimensional array of L sites with either open or periodic boundary conditions. Each site can be either occupied by a single vehicle or empty.

A vehicle on the array can have an integer velocity value between 0 and v_{max}.

To calculate a single time step, each vehicle on the array undergoes the following consecutive steps in parallel:
Acceleration, **Slowing down**, **Randomization**, and **Car motion**

Numerical Approach

The model to be constructed is called a ‘Boolean’ model where, if certain conditions are met, the behaviour of a cell will change.

For each time step and each car there are four things that need to be calculated.

First is acceleration: If the previous velocity ‘v_n’ of the car is less than v_{max} and if the distance between it and the next car is greater than $v_n + 1$, then the speed of the car is increased by 1 ($v_{n+1} = v_n + 1$).

Second is slowing down: If the car sees another vehicle j units in front and $j <= v_n$, then decelerate to $j – 1$ ($v_{n+1} = j – 1$).

Third, to simulate the random nature of humans at the wheel, is randomization: There is a probability ‘p’ that the velocity of the car will decrease by 1 ($v_{n+1} = v_n – 1$) if $v_n > 0$.

Numerical Approach (cont'd)

The fourth and final item is car motion: The car is moved v_{n+1} units forward

Through these four steps very general properties of traffic are modelled on the basis of integer valued probabilistic cellular automaton rules.

Visualization and Plotting Tools

- Will be mainly using MATLAB's plotting tools

Testing and Numerical Experiments

- Compute model for varying numbers of time steps and increasing vehicle densities with both periodic and open BCs

- Compare data to data from similar traffic models as well as real traffic data

- Investigate behaviour of the system when a bottleneck is introduced

Project Timeline

Dates	Activities
Oct. 21 – 31	Begin research and code design
Nov. 1 – 8	Implement code
Nov. 9 – 15	Test code
Nov. 16 – 25	Take and analyze data and begin report
Nov. 26 - 30	Finish report
Dec. 2	Submit report

References

Nagel, K., & Schreckenberg, M. (1992). A cellular automation model for freeway traffic. *Journale de Physique I*, *2*, France, 2221-2229.

gridlock.jpeg: http://www.cs.nyu.edu/courses/spring09/V22.0202-002/lectures/lecture-03.html

Diffusion Limited Aggregation

Phys 210 Term Project Proposal

Elyjah Kiyooka

Overview:

- Diffusion Limited aggregation is a simple model that can be used to understand the motions of zinc ions in an electrolyte, iron fragments attaching to a magnet, and other fractal structures
- The model is based on random motion of a particle on a grid (Brownian motion)
- This particle will move around in this random fashion until it comes to be adjacent to the default stationary particle in the center of the grid or moves off of the grid
- More particles are add individually once the last particle is attached or has moved off grid

Goals:

- Write a matlab code that can undergo this process for a large number of particles
- Test this code by analysis of mpeg file for desired behaviour and by comparing with known solutions
- Investigate the density of particles in a certain area to give you some measure on how the particles cluster

Project Formulations

- Particle will live in a 2D grid with indices (m,n) starting from the top left labeling each point in the grid
- One particle will start in the center if indices m,n even (m/2,n/2) or if m,n odd (m/2 + 1/2, n/2 + 1/2)

a_{11}	a_{12}	$\cdots$	a_{1n}
a_{21}	a_{22}	$\cdots$	a_{2n}
$\vdots$	$\vdots$	$\ddots$	$\vdots$
a_{m1}	a_{m2}	$\cdots$	a_{mn}

- Project formulations continued
- The added particles will start on some random point on the border of the grid found by a matlab random number generator
- Random number generates (1,2,3,4) to decide whether it will start on the top or bottom row or left or right column (eg. 1 will be m = 1 particle start in first row; 2 will be m = m particle will start in last column)
- Then random number generator will find a random number to determine what the other index will be (random number from 1 to m/n)

- Project formulations continued
- The particle will have four motions on the lattice (up, down, left, right) decided upon randomly using a matlab random number generator (eg. Up will be 1, etc.)
- Once the particle moves adjacent (to the left, right, top or bottom of another particle) then the particle's position on that those indices will be fixed and the program will then reiterate

Numerical Calculations

- Calculation of particle density:
- # of particles in area $\Delta m * \Delta n$ divided by area $\Delta m * \Delta n$

Visualization and Plotting Tools

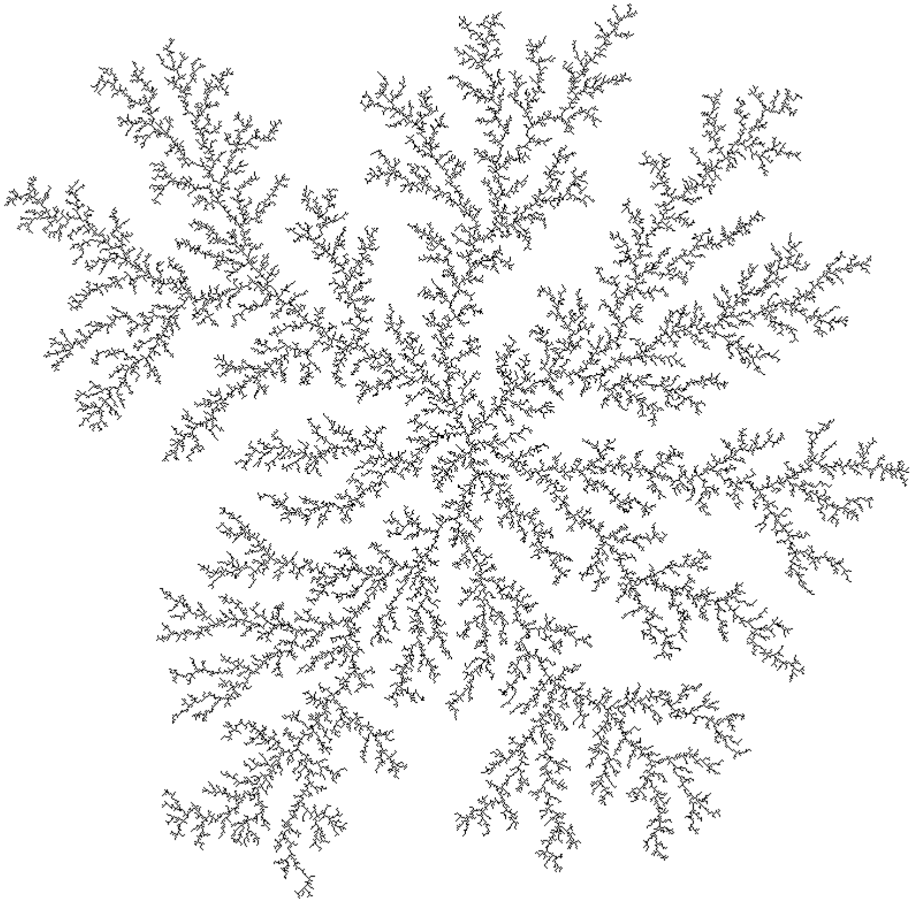
- I will use xvs for generation of mpeg animations, and will also use MATLAB plots for analysis and calculations for my report

Testing and Numerical Experiments

- Visual analysis of mpeg file to see if desired behaviour of diffusion and aggregation is acquired
- See if final result produces a fractal pattern as it should
- Comparison with other studies fractal pattern
- Comparison with other studies average of measurements of particle number in given area approximately match for different values of sticking constant

Project Timeline

Dates	Activity
Oct 23 – Nov 5	Write code, and begin research
Nov 6 - 15	Finalize code, and test. (hopefully remove all bugs)
Nov 16 – 25	Begin report, make calculations, analyze results
Nov 26 – Dec 02	Make comparison, finish report
Dec 03	Submit project



- References:

- Bourke, Paul. Diffusion Limited Aggregation. September 2014. http://paulbourke.net/fractals/dla/
- Choptuik, Matt. http://laplace.physics.ubc.ca/210/Doc/term-projects/kdv.pdf

- Questions ?
- Comments ?

Solving the Gravitiational N-body Problem With Finite Difference Methods

Karlo Krakan – 31451123
October 23, 2014

Project Overview

- The gravitational n-body problem is a problem in classical physics to determine the motion of 3 or more bodies interacting gravitationally
- Thus far, only n-body problems that have been solved analytically are the two body problem and the restricted three body problem. It is then of interest to solve larger n-body problems numerically

Project Goals

- Use MATLAB to simulate the n-body problem
- To test the simulation of the n-body problem with various initial conditions and with various n values
- To animate the simulation using animation software

Mathematical Formulation

- Newtons law of gravitation for n bodies

$$m_i \frac{d^2\mathbf{q}_i}{dt^2} = \sum_{j=1, i\neq j}^{N} \frac{G m_i m_j (\mathbf{q}_j - \mathbf{q}_i)}{\|\mathbf{q}_j - \mathbf{q}_i\|^3} = \frac{\partial U}{\partial \mathbf{q}_i}$$

- Where q_i is the position of the i-th mass
- 3n second order differential equations

Numerical Approach

- Solve the n-body problem with a three dimensional lattice and using finite difference methods to calculate future positions and velocities

- $f'(x_j) = f_{j+1} - f_{j-1} / 2\Delta x$

Testing

- Will test with various n
- Will test with various initial conditions
- Will test with various lattice sizes
- Will test whether the conservation laws hold in the model

Project Timeline

- October 23 – November 7
  - Research, formulate FDAs, begin designing code
- November 7 – November 14
  - Finish designing code
- November 14 - November 24
  - Test code, run simulations, analyze data
- November 24 – December 3
  - Write and submit report

References

- Newtons equation from
  - http://en.wikipedia.org/wiki/N-body_problem

Gravitational N-Body Problem

CHAPMAN KWAN

PHYS 210 - TERM PROJECT PROPOSAL

Overview

- N-body problem is a problem that uses differential equations and other mathematical formulations connected with physics, such as Newton's Principia, and general relativity, to predict the motion of groups of celestial bodies.
- One of the most awesome component to the N-body problem is gravity, where Newton expressed in terms of differential equations.

Project Goals

- Use of simulation, through MatLab programming, to solve the problem of gravitational motion between several bodies under this force
- Understanding the codes produced off of MatLab and having it actually produce the simulation as stated above
- Using FDAs to determine a code for the simulation

Mathematical Formulation

- Simple equation relating to the N-body problem

Newton's gravity equation: $F = \frac{Gm_1m_2}{r^2}$

G is the gravitational constant = $6.67384 \times 10^{-11}\ \mathrm{m^3kg^{-1}s^{-1}}$

m is the mass of the particles

and r is the radius between the two particles

Equations for 2-Body p

$$m_1a_1 = \frac{Gm_1m_2}{r_{12}^3}(r_2 - r_1)$$

$$m_2a_2 = \frac{Gm_1m_2}{r_{21}^3}(r_1 - r_2)$$

Mathematical Formulation

- By finding the acceleration by rearranging below

$$\vec{F} = \frac{Gm_i m_j}{(r_i - r_j)^2}\hat{r} = m\vec{a}$$

We get that acceleration is,

$$a = \frac{Gm_j}{(r_i - r_j)^2}\hat{r} \text{ in direction, } \hat{r} = \frac{(r_i - r_j)}{|r_i - r_j|}$$

After some moving around we get that…

$$a = \frac{d^2 r_i}{dt^2} = \frac{Gm_j(r_i - r_j)}{|r_i - r_j|^3}$$

Numerical Approach

- By using FDAs, finite difference approximations, while changing the points for time and displacement, (time=t, displacement=x) in order to find points within a set.
- Each point for time and displacement will be separated in set increments of Δt and Δx
- From lectures and knowing that

$$a(t_i) = x''(t_i) \cong \frac{x(t_i + dt) - 2x(t_i) + x(t_i - dt)}{dt^2}$$

where $dt = \frac{t_{final}}{n_k - 1}$, and $t^n = (n_k - 1) * dt$, and $n = 1,2,3,4 \ldots n_k$

Visualization and plotting

- Through MatLab and it's genius plotting skills, we can visualize the N-bodies

Testing & Numerical Experiments

- Use different masses for several particles which are non-zero in magnitude
- Using standard values such as t=0 and x=0 for initial points
- Using different starting velocities
- Testing with different sets of N-bodies

Project Timeline

Dates	Type of Work
Oct 17 – Oct 23	Research and mathematical derivation
Oct 23 – Oct 29	Begin coding/programming
Oct 30 – Nov 19	Implementing Code
Nov 20 – Nov 21	Code testing
Nov 22 – Nov 27	Running numerical experiments, analyze data and start the project report
Nov 28 – Dec 01	Complete and finalize report
Dec 01 – Dec 02	Submit project (Not on Dec 03! Too Risky!)

References

- Some references from previous years of presentations, and using Matt's sample one to basically format it in a similar way
- http://en.wikipedia.org/wiki/N-body_simulation
- http://physics.princeton.edu/~fpretori/Nbody/intro.htm
- http://www.scholarpedia.org/article/N-body_simulations_(gravitational)

- Note that, the mathematical formulation slide is a combination of everything but then its equivalent

General Gravitational N-Body Problem

Physics 210 Term Project Proposal

Kevin Kwon

October.18/2014

Overview

- In an N-Body problem, N number of bodies interact with each other through forces.
- The N-Body model can simulate how point masses interact each other through gravity in various different initial states.
- In my project, I will be dealing with gravitational forces between bodies

Project Goals

- Use MATLAB to write a code that can solve a gravitational N-Body problem using second order finite difference methods.
- Test my MATLAB code to make sure it actually works
- Try out different initial conditions

Mathematical Formulation

- Newton's equations of gravity: $$\vec{F} = \frac{Gm_i m_j}{r^2}\hat{r} \qquad \vec{F} = m\vec{a}$$
- Solve for a_1 → $\vec{a}_i = \frac{\vec{F}}{m_i} = \frac{Gm_j}{r^2}\hat{r}$ where r can be expressed as r_i-r_j. $\hat{r}$ is the unit vector i.e. the direction and $$\hat{r} = \frac{(r_i - r_j)}{|r_i - r_j|}$$
- therefore the equation becomes $\vec{a}_i = \frac{Gm_j(r_i - r_j)}{|r_i - r_j|^3}$ and $a_i = \frac{d^2\vec{x}_i}{dt^2}$
- Finally if we replace r with x we get something beautiful

$$\frac{d^2 x_i}{dt^2} = \frac{Gm_j(x_i - x_j)}{|x_i - x_j|^3}$$

Numerical Approach

- I will be using a second order finite difference technique, where I replace all the continuous time and space (t,x) with a discrete set of points (t^n,x_i)
- Each t and x will be separated by Δt and Δx respectively.
- I will use $a(t_i) = x''(t_i) \cong \dfrac{x(t_i + \Delta t) - 2x(t_i) + x(t_i - \Delta t)}{\Delta t^2}$ and $\Delta t = \dfrac{t_{\max}}{(n_t - 1)}$

$$t^n = (n_t - 1)\Delta t \;\; , \;\; \mathrm{n} = 1,2,3,\ldots,\mathrm{n_t} - 1, n_t$$

t^n is the n^{th} time, and n_t is the number of grid points I will use.

Testing

- Test with different Δt
- Test different numbers of N bodies
- Test different initial velocities

Project Timeline

Date	What to do
Oct.18~Oct.27	Do some research. Think about how to code
Oct.28~Nov.16	Write code
Nov.17~Nov.20	Test Code
Nov.21~Nov.25	Run experiments with code, obtain and analyze data, start report
Nov.26~Nov.28	Try to finish up report
Nov.28~Dec.3	Hand in somewhere around here

Comments?
Suggestions??
Questions???
(give me some suggestions please so I can do good)

Thank you

For listening to me talk

References

- http://en.wikipedia.org/wiki/Newton's_law_of_universal_gravitation
- Various previous presentation slides
- http://blogs.discovermagazine.com/d-brief/2013/06/20/applause-contagious-like-a-disease/

Simulation of a Simple Neural Network

PHYS 210 TERM PROJECT PROPOSAL

IVAN LAN - 37738135

Overview

A neural network is composed of approximately 10^12 neurons that are connected to one another and communicate by sending, or "firing" electrical pulses. In regards to this project, we are assuming that any given neuron is either firing or not firing.

The neurons' spin configurations, as in whether a neuron is firing or not, can be represented by a lattice which can represent a pattern in a subjects memory

Project Goals

- To be able to store and represent desired patterns of spins using a two-dimensional array
- Additionally store the energy used by neurons to interact with one another

Mathematical Formulation

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

- $J_{i,j}$ refers to the strength of the connections between neuron i and neuron j while the sum represented is takes in all pairs or i and j in the network.
- Whether the energy is positive or negative also determines whether a neuron has a positive or negative firing rate

Mathematical Formulation (continued)

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

- m and n are two different patterns, N is the total number of spins in a certain configuration
- Used to factor in memories

Numerical Approach

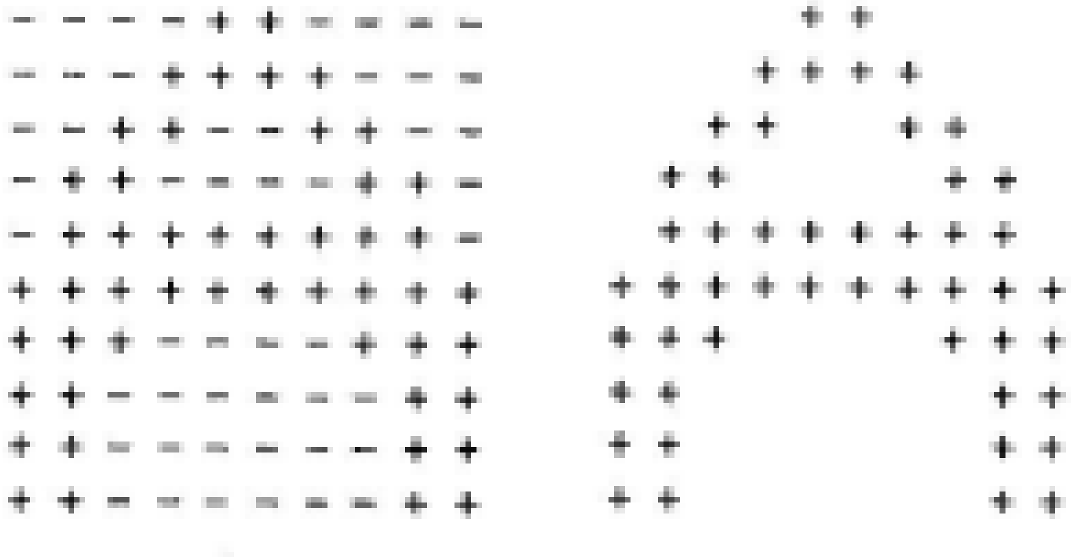
- The Monte Carlo method is used alongside the energy function to determine how the spin system changes over time
- This method has us calculate the energy to flip a certain spin in some particular configuration of a network

Visualizing and Plotting Tools

I will primarily be using MATLAB in order to generate an array to illustrate the neural network

(- spins removed to more easily

see the desired pattern)



Testing and Numerical Experiments

I will test out various pattern to determine if they are being accurately stored by the program

Project Timeline

Oct 20-26: Research and beginning code design

Oct 27-Nov 2: Programming

Nov 3-9: Finish programming (productivity may decrease due to various midterms)

Nov 10-16: Error testing and begin analyzing data for report

Nov 17-23: Work on report

Nov 30: Finish report

References

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

Questions?

Finite difference solution of the one-dimensional time-dependent schrodinger equation

Phys 210 Term project proposal Yifan Li

Overview:

The Schrödinger equation is a partial differential equation that describes how the quantum state of a physical system changes with time.
In the standard interpretation of quantum mechanics, the wave function is the most complete description that can be given to a physical system. Solutions to Schrödinger's equation describe not only molecular, atomic, and subatomic systems, but also macroscopic systems.

Project goals:

To write an MATLAB (octave) code which solves the TDS equation numerically, using second-order Finite difference techniques

To establish correctness of the implementation of the code through convergence tests and comparison with known solutions

To investigate a variety of initial conditions for the equation, like those describing different initial potential functions.

Mathematical formulation:

The TDS can be written in the form

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

On the domain $0 \leq x \leq 1 and t \geq 0$

And after setting h/2pi =2m =1 we get

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

and with initial potential functions:

V(0,x)=Vo(x)

and with initial wave function:

$\psi(0,x) = \psi_0(x)$

Numerical approach:

Then using the Crank Nicholson approximation which is firstly second order in both space and time and secondly implicit meaning a system of linear eqns must be solved at each time step:

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

Where $\psi_j^n = \psi(j\Delta x, n\Delta t), V_j = V(j\Delta x); x_j = 0 + jh, j = 1,2,3\ldots n_x; t^n = n\lambda h, n = 0,1,2\ldots n_t$

And rewrite the above eqn in the form :

$$c_j^+\psi_{j+1}^{n+1} + c_j^0\psi_j^{n+1} + c_j^-\psi_{j-1}^{n+1} = S_j$$

Numerical approach(continued)

And with

$$c_j^+ = \frac{1}{2\Delta x^2} = c_j^-; c_j^0 = \frac{i}{\Delta t} - \frac{1}{\Delta x^2} - \frac{V_j}{2}$$

And therefore also,

$$S_j = (-c_j^+)\psi_{j+1}^n + (-c_j^0)\psi_j^n + (-c_j^-)\psi_{j-1}^n$$

With possible boundary condition for an infinite potential well:

$$\psi_1^{n+1} = \psi_{nx}^{n+1} = 0$$

Which constitute a complex tridiagonal linear systemfor the advanced valve of the wave function.

Then use complex*16 arithmetic, and the LAPACK solver zgtsv to solve the tridiagonal system.

Testing and Numerical Experiments

Tesing :

- convergence test :
- Based on the normalization principle:

$$I = constant = \sum_{j}^{nx-1} \frac{1}{2}(\psi_j^n + \psi_{j+1}^n)\frac{1}{2}(\psi_j^{*n} + \psi_{j+1}^{*n})\Delta x \approx \int_0^1 \psi(x,t)\psi^*(x,t)dx$$

- fix initial data, compute solutions using discretization scales h, h/2, h/4…and ensure that O(h^2) convergence behavious is continued.
- Numerical Experiment:

  Still thinking about this point yet……

- 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.

- References:

http://laplace.physics.ubc.ca/210/Term.html/schrodinger.pdf

- Thanks for your comments and questions!

SIMULATION OF A SIMPLE NEURAL NETWORK

PHYS 210 TERM PROJECT PROPOSAL

OCTOBER 23, 2014

KRISTINE LOUIE

OVERVIEW

- What is a neural network?
  - Soma
  - Dendrites
  - Axon

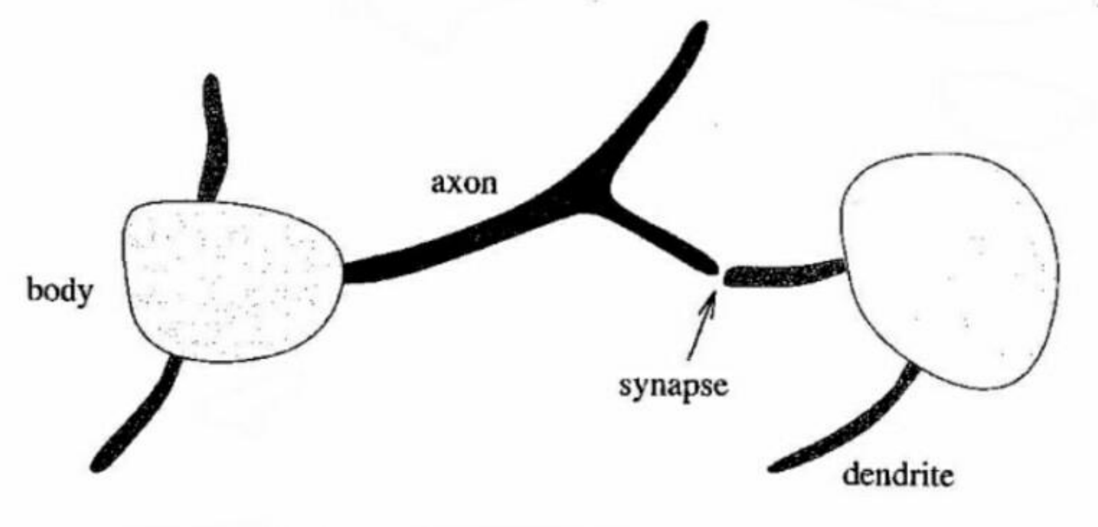


Figure 1: Basic structure of two neurons
(credits: http://laplace.physics.ubc.ca/210/Doc/term/Giordano-12.3-4-Neural-Networks.pdf)

PROJECT GOALS

- To model the interactions of a simple neural network
- To test how a neural network acts as a memory

MATHEMATICAL FORMULAE

- Firing Rate

  $R = f(\Sigma V_i)$ where V_i is the input signal from dendrite *i*

- Energy of a Neural Network

  $E = -\Sigma J_{i,j} s_i s_j$ where $\Sigma J_{i,j} s_j$ is the sum of the inputs to neuron *i*

GENERAL APPROACH

- Ising Model
  - Using a spin to represent a neuron, can be used to represent the interactions between neurons
- Monte Carlo Method
  - Given sets of numbers (inputs to the neuron), can be used to solve the Ising model through iterations

PROJECT TIMELINE

- Oct 20 – Oct 27: Research and beginning of code design
- Oct 28 – Nov 11: Creation of code
- Nov 12 – Nov 19: Test code (and fix as necessary)
- Nov 20 – Nov 28: Run experiments, analyze data, start report
- Nov 28 – Nov 30: Finish report
- Wednesday, December 3, 9AM: Submission deadline!

REFERENCES

- http://laplace.physics.ubc.ca/210/Doc/term/Giordano-12.3-4-Neural-Networks.pdf
- Wittwer, J.W. "Monte Carlo Simulation Basics." *Vertex42.com*, June 1. 2004. Web. Oct 18. 2014. <http://www.vertex42.com/ExcelArticles/mc/MonteCarloSimulation.html>

THANK YOU!

Gravitational N-Body Dynamics Simulation

Savannah Pulfer

Overview

- This simulation, which will be done in only 2 dimensions, demonstrates the interactions of n bodies ("particles").
- These bodies interact through only the force of gravity and the simulation aims to predict the positions and velocities of the particles based on the initial conditions.
- For systems with more than two bodies, this can only be approximated.

Project Goals

- Write a Matlab code to simulate the interactions of n bodies/particles and to predict their positions and velocities at some later time based on some specific initial conditions.
- Run this simulation with a variety of initial conditions.

Mathematical Approach

- Newton's Law of Universal Gravitation
  - $F = \frac{-GmM(r-R)}{|r-R|^3}$
- Newton's 2nd Law
  - $F = ma$
- Superposition Principle
- Other relevant kinematics

Numerical Approach

- Finite Difference Approximation:

$$\frac{F}{m} = f'(v_0) = \frac{f(v_0 + \Delta t) - f(v_0)}{\Delta t}$$

Testing and Numerical Experiments

- Test whether the code is working
  - Attempt with 2 bodies
  - Assure conservation of energy and momentum

- Numerical Experiments
  - Run simulation with various initial conditions
  - Run simulations for various lengths of time

Project Timeline

October 23rd	Present proposal
October 24th - November 6th	Write code
November 7rd – November 11th	Test code
November 11th – November 20th	Run numerical experiments
November 21st – November 26th	Analyse data
November 27th – December 1st	Write report
December 2nd	Submit report

References

- http://www.physicsclassroom.com/class/circles/Lesson-3/Newton-s-Law-of-Universal-Gravitation
- http://www.sparknotes.com/testprep/books/sat2/physics/chapter17section4.rhtml
- http://laplace.physics.ubc.ca/210/Proposals-2013/L1B.pdf

Toomre Model of Galaxy Collisions

PHYS 210 Term Project Proposal

Syed Nayyer Raza

Overview

- The Toomre brothers conducted the first simulations of galaxy mergers in 1970s, using a small number of particles in the simulation.
- Their model also made some further simplifications based on the fact that the mass of the centre of the galactic nuclei was much greater than the individual masses of the stars:
  - The individual stars do not exert any gravitational forces, only experience gravitational force from the galactic nulcei
  - The individual stars do not collide with each other, and can “pass through” one another
  - Dark matter, dark energy, and interstellar medium can be ignored
  - Newtonian mechanics are sufficient for the approximations.

Project Goals

- To write a MATLAB code that simulates the collission of two galaxies based on the Toomre model.
- Create a visual simulation of the galactic collision.
- Use various initial conditions such as position, velocity, angle of approach, mass of galaxy to explore their individual effects on the outcome of the simulation.
- Confirm that the model works within Physical limits, i.e. it observes the law of conservation of energy.
- To see whether the shape of the galaxies effects the shape of the outcome.
- To (try) to simulate the eventual collision of Andromeda and Milky Way.

Mathematical Formulation

- Newton's Law of Motion: F = ma = mv^2/r
- Newton's Law of Universal Gravitation: F = Gmm/r^2
- Kepler's Third Law: P^2 = 4pi^2r^3/Gm

Numerical Approach, Experiments, and Testing

- Use differential equation of velocity and acceleration.
- Discretize equation of motion using FDAs
- Define initial conditions.
- Redo until satisfied with the result, checking with established models.

Timeline

- October 14 - 21: Begin Researching and designing code
- October 21- November 10: Implement code
- November 10 - 15 : Test code
- November 15 – 25: Run numerical experiments, begin report and analyze data
- November 25 – 30: Finish Report
- December 1: Submit report

Finite Difference Solution to N-Body Problem

Physics 210 Term Project Proposal

Chris Scott

October 2014

Overview

- System of N bodies interacting through gravitational field
- Each body exerts gravitational pull on all other bodies and experiences pull itself
- 2 spacial dimensions (x, y) and 1 temporal dimension (t)

Project Goals

- To construct a simulation of how independent particles interact in a gravitational field
- To explore a range of initial conditions (varying distances, configurations) which include particles starting at rest and starting in motion
- To consider effects of non-identical masses
- To have fun

Mathematical formulation

- Newton's law of gravitation:
- $$\frac{\left(d^2 \vec{x}_i\right)}{dt^2} = G \sum_{k=1}^{n} m_k \frac{\left(\vec{x}_k - \vec{x}_i\right)}{\left(\vec{x}_k - \vec{x}_i\right)^3}$$

$$\frac{\left(d^2 \vec{x}_i\right)}{dt^2} \stackrel{\text{def}}{=} a_i$$

Domain:

$$0 \leqslant t \leqslant t_{max}$$

$$-x_{max} \leqslant x \leqslant x_{max}$$

$$-y_{max} \leqslant y \leqslant y_{max}$$

Initial conditions (also varying m):

$$\frac{\left(d\vec{x}\right)}{dt} = 0$$

$$\frac{\left(d\vec{x}\right)}{dt} \neq 0$$

Numerical Approach

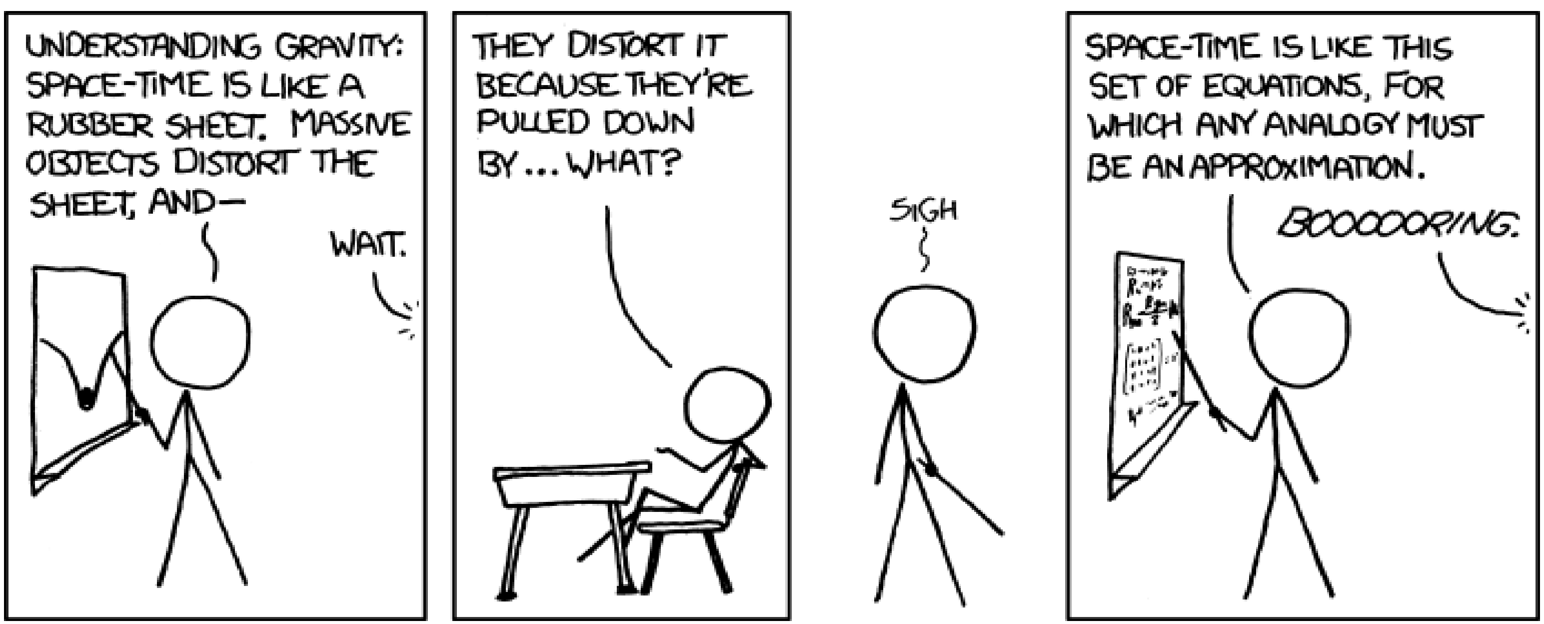
- The problem will be solved by converting the equation of motion into a 2nd order finite difference approximation

- Replace continuum with discrete lattice:

$$t_i = ih, i = 0, 1, 2, \ldots, n_t$$
$$x_j = -x_{max} + jh, j = 0, 1, 2, \ldots, n_x$$
$$y_k = -y_{max} + kh, k = 0, 1, 2, \ldots, n_y$$

Testing

- Crucial to ensure that discrete equations converge, so testing will be done at intervals h, h/2, h/4, … until convergence to 2nd order in h is established



Project Timeline

Oct 20 – 25: Preliminary research, equation derivations and code design

Oct 26 – Nov 17: Code implementation

Nov 18 – 22: Code testing

Nov 22 – 28: Data analysis and report writing

- Nov 30: Submission

References

- Equations taken from

  http://en.wikipedia.org/wiki/N-body_problem#n-body_ choreography

- XKCD comic from google images

Simulation of Equilibrium Configuration of N-Identical charges

Earl Tabones

Overview

- This is a Simulation of N particles interacting with each other.
- Charges interact by Coulomb's Law (Like charges repel, Opposite charges attract)
- Equilibrium is obtained once charges are stationary and net forces on each particle is zero.

Goals

- To create a visual representation of N-body interactions and the movements of particles
- To create a code in Matlab(Octave) which simulates the problem.
- To investigate the motion of the particles in initial conditions.

Assumption

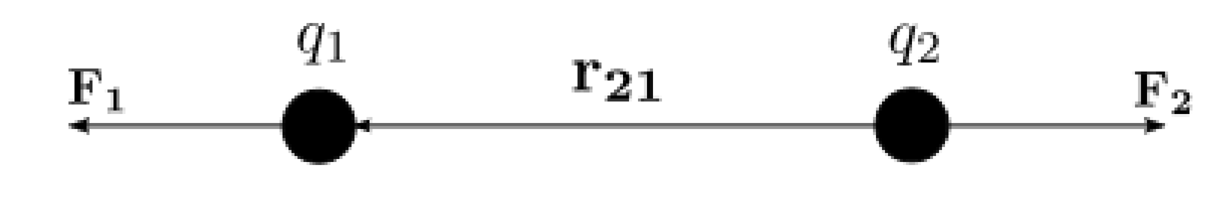
- Each body has an Initial state
- There are no outside interactions.
- The bodies will be assumed to be a concentric Sphere.

Mathematical Stuff

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

$$\mathbf{F}_1 = k_e \frac{q_1 q_2}{|\mathbf{r}_{21}|^2}\hat{\mathbf{r}}_{21}$$

Charles-Augustin de Coulomb



Project Timeline

DATES:	ACTIVITIES
Oct 20 – Oct 27	Research and Design Code
Oct 27-Nov 1	Implement Code
Nov 1-Nov 8	Run numerical experiments, work on presentation & report
Nov 8-Nov 22	Analyze data and complete report and presentation
Nov 22-Nov	Present Project
	Submit Report

References

- http://en.wikipedia.org/wiki/Coulomb's_law (AKA best site on earth)

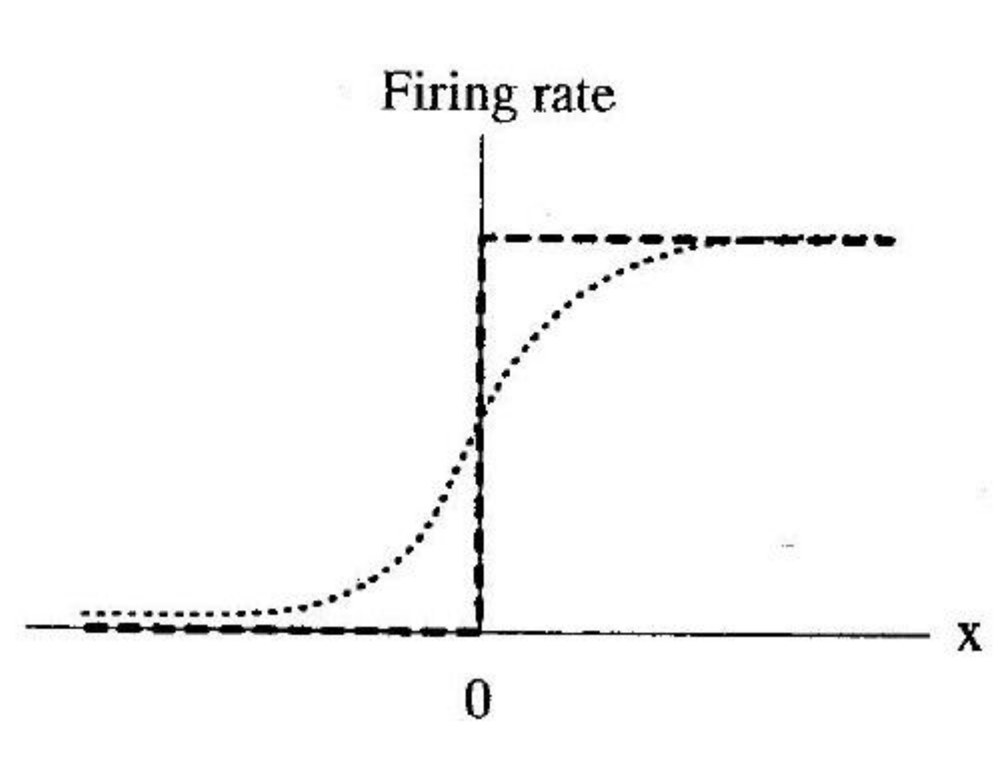
Me-Me-Me-Memory Creation and Lost using *The Ising Model*

A Physics 210 Production

Timothy Tan CG

The Overview

- Memory : Interaction of neurons
  - Electrical Pulses
  - Either "firing" at a **high** rate or **low**.
  - Like an "**up**" spin or a "**down**" spin.
- Ising Model : Loads of simple units with up/down spins
  - Each spin **effects the other units** around them
  - Allows for large-scale interaction between units

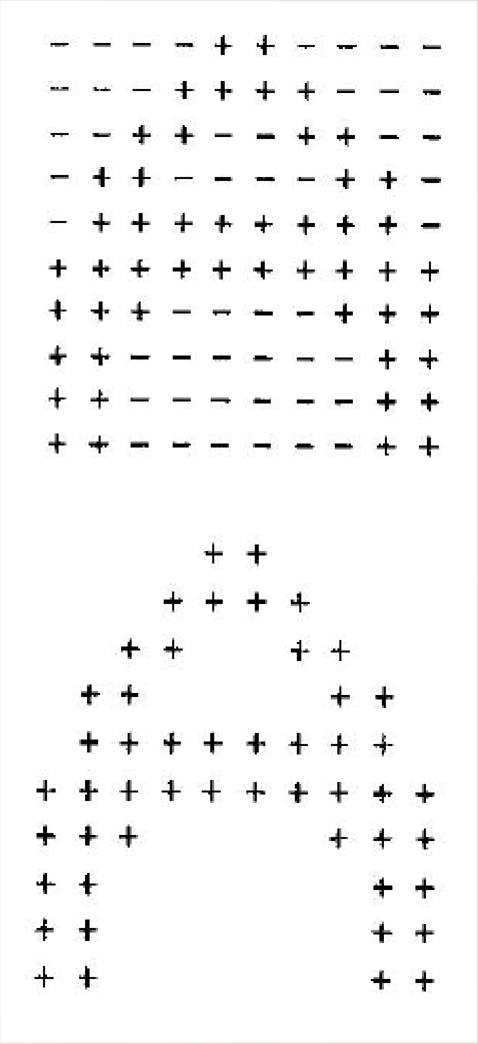


Goals

- *Initially,* to **reproduce** the Ising Model as devised in the reference material using MATLAB. A success model should:
  - Visually model *memory* using a *mxn* array with spins that arranges itself into a recognizable **symbol** (to us) – like an "A" or "B".
  - When given a pattern similar to a *remembered* symbol, **recreate** that symbol through transforming the spins into the symbol.
- To measure tolerable level of **memory loss** this model simulates by removing spins in the *recognized* symbol and re-trying the test.
- *BONUS*: If I get to it, to recreate the part of the model that creates new memories and models memory fading over time.

“Memory” according to the Model

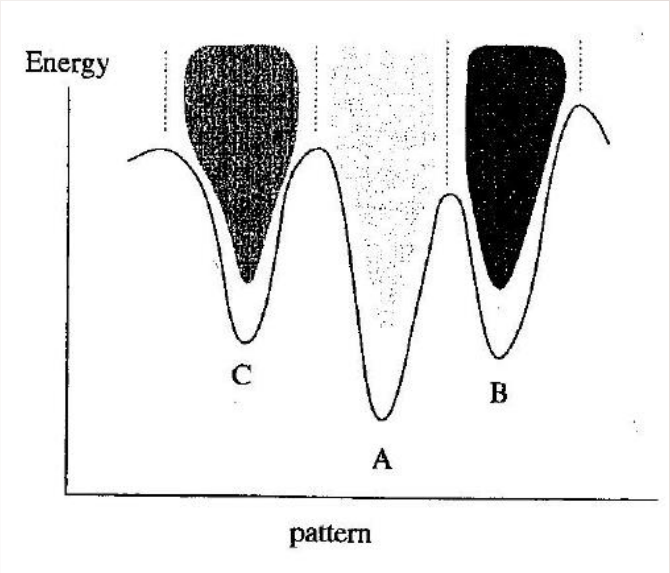
- **Spins** on units depend on its energy E
  - $E = -\sum_{i,j} S_j S_i J_{i,j}$
  - When $\sum_j \; S_j J_{i,j}$ gives a negative E, neuron *i* will likewise have a negative value: $S_i = -1$, and vice-versa.
    - S_i being the spin value, and $J_{i,j}$ being *interaction energy*.
- ***Memory** is held when spins are in a desired **order**.*
  - One flip will be chosen to start, then:
  - If $\Delta E_{flip} \geq 0$ then the unit doesn’t flip, *vice versa*.
- Ising **model** memory is *shaped*.
  - Using 10 x 10 array of units of + or – spins.



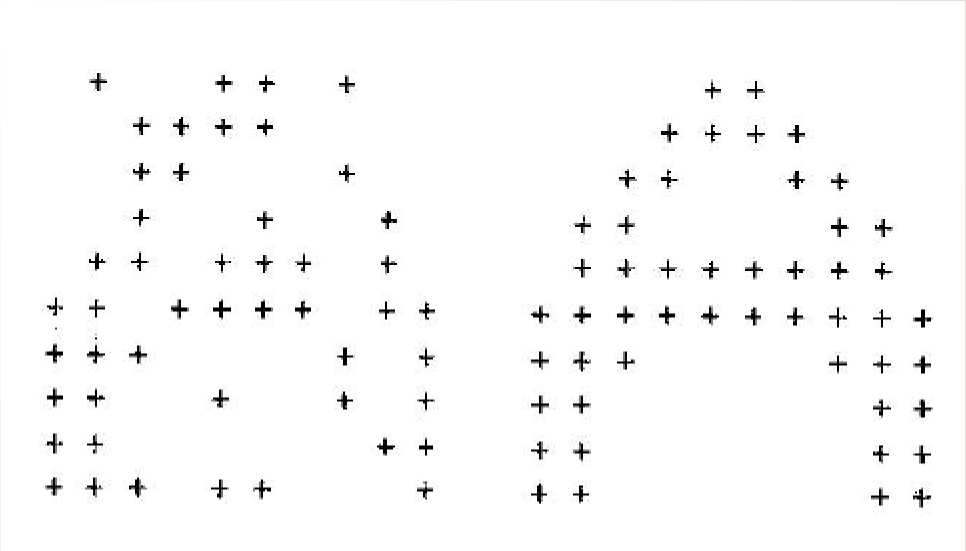
Energy Diagram

- "Basin of Attraction" (the minima of recognized patterns are supposed to draw the minima slope of imputed patterns towards it, therefore recreating the recognized pattern.)

- Distance between two patterns (actual and memory):
  - $\Delta_{m,n} = \frac{1}{N}\sum_i [S_i(m) - S_i(n)]^2$
  - M and n are different patterns, N total number of spin, Si the spin location
  - If distances between patterns are large, then signals are easier to distinguish



Energies of recognized symbols(A, B, C) plotted against pattern imputed.



Interaction Patterns

- To bring spins into the correct pattern : E must be negative, so *interaction energies* must be fitted as such.
  - $J_i = S_i(m)S_j(m)$ (credited to *Hebb* and *Copper*)
  - So: $E(m) = -\sum_{i,j} J_i S_i S_j = -\sum_{i,j} S_j(m) S_i(m) S_i S_j$
  - **note:** if imput is *too* random, E~0 thus no change will happen.
  - **note:** m, n represent rows and columns while i, j are a numbering scheme for the spins

- Storing multiple patterns requires another equation:
  - $J_{i,j} = \frac{1}{M} \sum_m S_i(m) S_j(m)$
    - m refers to stored index, M is total patterns.
    - Network has N spins ~ and so N^2 different values of interaction energies $J_{i,j}$

Maximum Memory

- Memory in model is limited by patterns being **too similar** to one another.
  - *Best to have memorized patterns be as orthogonal as possible*.
  - Tests show that around ~0.13N patterns is when model starts destabilizing
- **Damage testing**: randomly set some values in $J_{i,j}$ matrix to zero
  - Tolerable damage depends on number of memories stored
  - Can see effects by running experiment and observing how often the pattern is 'recognized'.

Learning (and extra notes)

- Simple procedure for learning:
  - $J_{i,j}(new) = \beta J_{i,j}(old) + \alpha S_i(p)S_j(p)$
  - Si(p) is new pattern, **alpha** is parameter (controls how fast learning happens), **beta** -> adjusted to allow for fading of old memories
- Notes:
  - $i = N(m-1) + n$
    - **Note:** above mapping can be inverted
  - **Thus**: storage requires Array of size N^2 x N^2
    - 10x10 array requires 10^4 interaction energies
  - Recognized patterns are all imputed manually.

Testing & Numerical Experiments

- **Initially**: After figuring recognized patterns into the model, I'll place spin patterns that are similar in shape to one of the recognized to see if the model will transform it into the recognized pattern.
- **Limits:** Finding the limits of the model by:
  - having about or more than 0.13N number of recognized patterns
  - Having patterns being similar to each other
  - Randomized imput pattern
- **Damage:** Changing a few spins in the recognized pattern to simulate how the model deals with memory loss: Will imput more similar shapes to find if the model finds their patterns.
- **Training**: Implement learning/fading equation to try to simulate the learning and losing of memory.

Project Timeline and Reference Source

Dates	Activities
20/10 – 26/10	Research equations, implementation for project & start coding
27/10 – 2/11	Implement code
3/11 – 9/11	Test code
10/11 – 16/11	Run numerical experiments, analyze data, begin report
17/11 – 25/11	Finish report
26/11	Submit final project! (due date is Dec 3rd)

- *References*

  - Giordano. "Neural Networks and the Brain." (n.d.): 418-39. Print.

Modeling N-body gravitational interactions

WITH USE OF THE TMOORE MODEL

Overview

- Collisions between galaxies radically change the shape they form.
- The Toomre Model, is a fairly simplified approximation of the interactions which occur. Ignoring dark matter, and any interstellar medium.
- The celestial objects are represented as particles, mass relative to size.

Objectives

- To create a simulation of two galaxies colliding where one is significantly larger than the other (Milky way << Andromeda) using the conditions of the Toomre model.
- To present the results in a visual manner. i.e. .mpeg, .gif, .jpg.

Mathematical Formulae

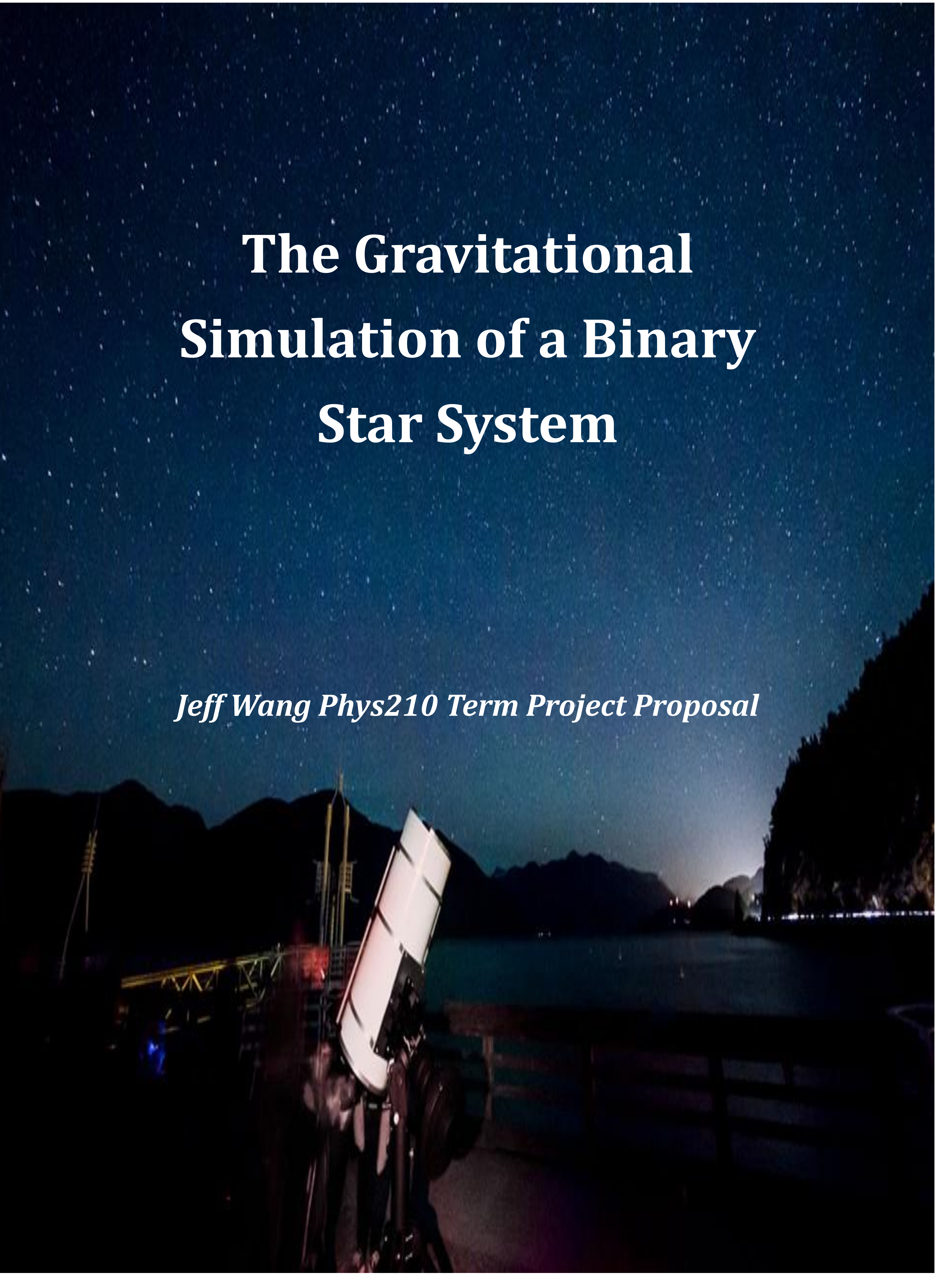
- $F = G\frac{m_1 m_2}{r^2}$
- $F_c = ma_c = \frac{mv^2}{r}$
- $P^3 = \frac{(4\pi^2)a^3}{G(M_1+M_2)}$

Timeline

- 16/10 – 22/10
  - Initial research and code designing
- 23/12-1/11
  - Implementing code
- 1/11-15/11
  - Testing, Debugging and Collecting data
- 16/11-29/11
  - Writing up report, and editing code for readability, and efficiency.

References

- Alan Toomre. Wikipedia. http://en.wikipedia.org/wiki/Alar_Toomre.15/10/14
- Toomre Sequence. COSMOS - The SAO Encyclopedia of Astronomy. http://astronomy.swin.edu.au/cosmos/T/Toomre+Sequence. 16/10/14
- Spiral Galaxies. KDE. https://docs.kde.org/stable/en/kdeedu/kstars/ai-spiralgal.html. 18/10/14



The Gravitational Simulation of a Binary Star System

Jeff Wang Phys210 Term Project Proposal

Project Overview

Why binary star system?

Because about half of the observed star systems are binary, they are very common in our universe!!

It is very special for us to imagine seeing two suns in the sky

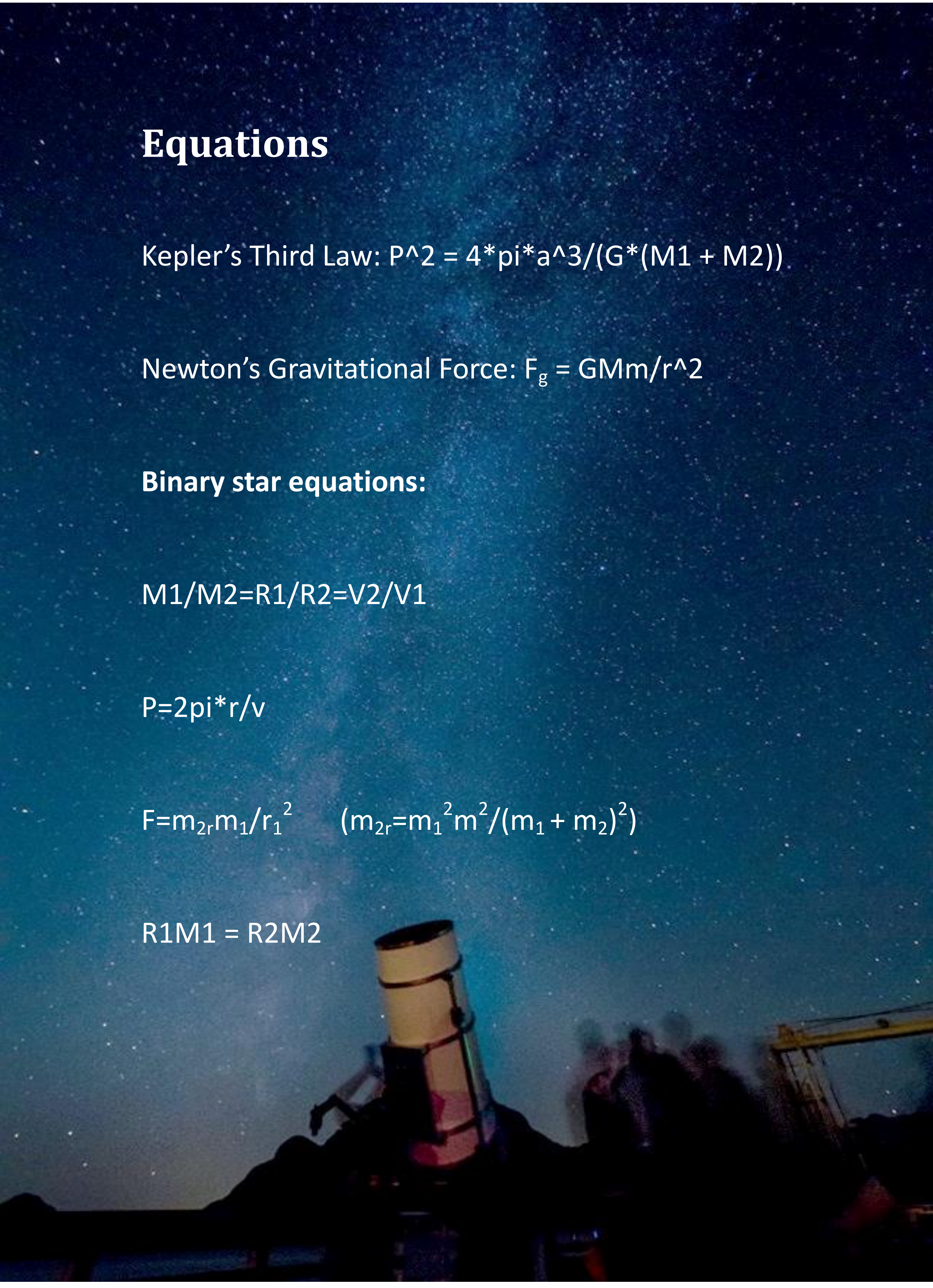
It will be take quite a bit of imagination to picture a world with two suns in the sky, but in fact astronomers think there is a good chance that a close binary system will support life, such as Kepler-47, it lasts longer (smaller mass) and it also has a broader habitable zone for life to develop (more "gentle" radiation).

Objective

- To understand and simulate the motions of a stable binary star system with MATLAB (varying the M1/M2 proportion)

- To code the simulation about how will planets orbit these binary star systems

- To investigate if a three body star system is possible? (known to be unstable)



Equations

Kepler's Third Law: P^2 = 4*pi*a^3/(G*(M1 + M2))

Newton's Gravitational Force: F_g = GMm/r^2

Binary star equations:

M1/M2=R1/R2=V2/V1

P=2pi*r/v

$F=m_{2r}m_1/r_1^2$ $(m_{2r}=m_1^2m^2/(m_1+m_2)^2)$

R1M1 = R2M2

MATLAB approaches

Using FDA and Tylor's approximation get the gravitational force and other factors such as speed, period and radius.

Use grid points and grid functions to achieve the approximation.

Update the position accurately and frequently to allow the simulation looks smooth

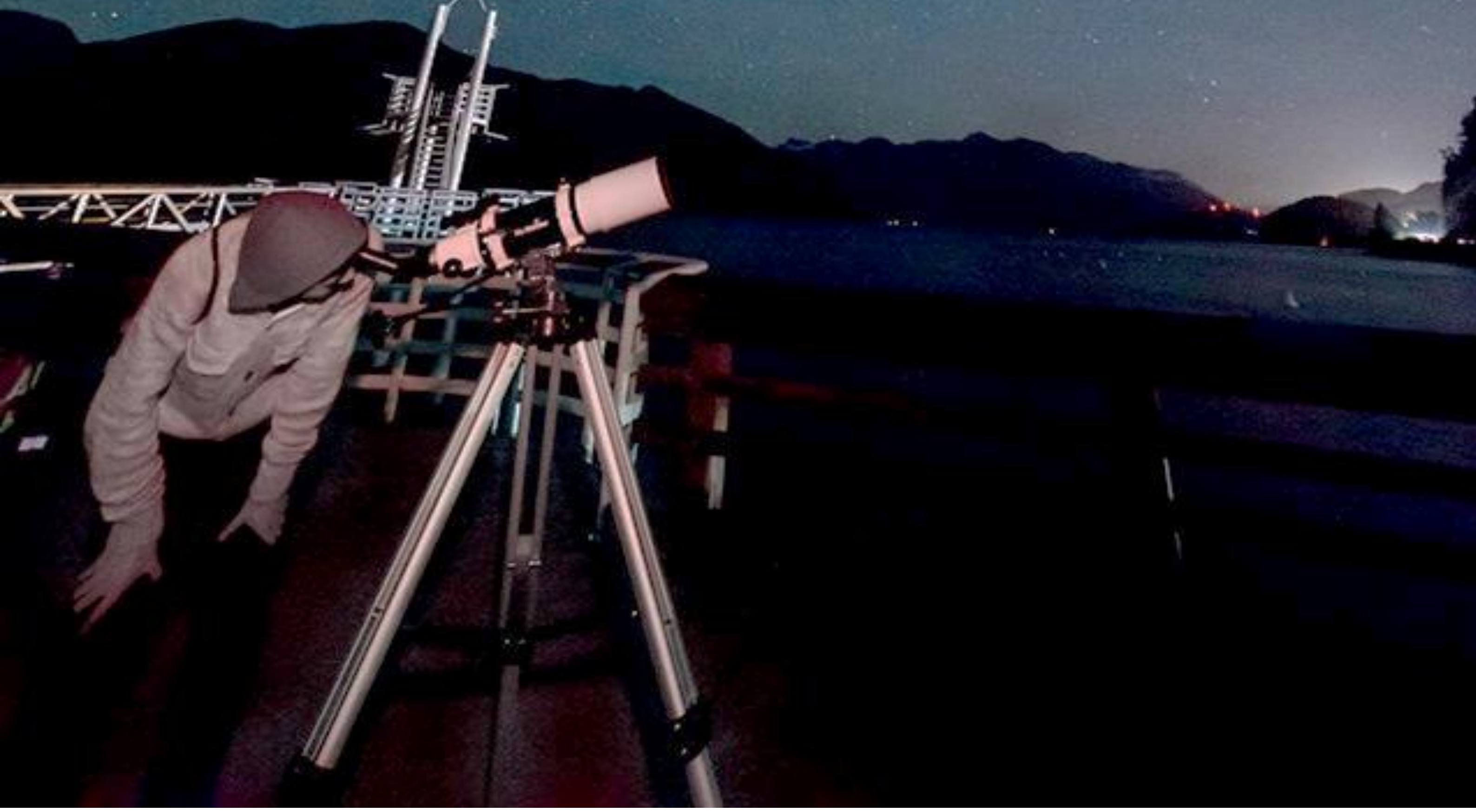
MAYLAB plotting will be used for simulation (in 2D)

Condition setting

- First simulate a simple binary star system, while experimenting different mass ratio

- Then I plan to add in lower pass bodies (planets) to orbit the system, will try to add in a various planets with different masses and initial conditions

- Finally experiment if a three-star-system is possible, can strip away planets first, starting with equal mass, I will then vary the mass proportion.



Timeline:

~ 11/4	Research coding ideas and proper ways to do the simulation
11/6~11/18	Finish Coding, and complete testing/debugging
11/20~11/27	Produce report.
12/2	Class presentation

Reference:

Habitable Binary Star Systems-AstroBiology Magazine

http://www.astrobio.net/news-exclusive/habitable-binary-star-systems/

2012/2013 Previous presentations

http://laplace.physics.ubc.ca/210/

Astro 200 Class notes- Professor Paul Hickson 2013

From connect webpage

Photos: UBC Astronomy Club (2014 Porteau Cove trip)


¿Questions?
¿SUGGESTIONS?


Thank you!!
Merci!!
Gracias!!
謝謝!!
Спасибо!!

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

PHYS 210
Aslan Zarei
(35141143)

Overview

- Motion of n-body particles can be approximated by gravitational forces between them.

- Gravitational N-body simulations, that is numerical solutions of the equations of motions for N particles interacting gravitationally have applications from few body to galactic and cosmological scales.

Project Goals

- To solve the n body problem using MATLAB and the related equations.

- To simulate particle's motion in 2D or even 3D space

- To test the code and verify its validity by applying different initial values and law of conservation of eenergy using FDA method

- To use a software to visualize interacting celestial objects.

Mathematical Formulation

Newton's Law of Gravity:

$$\vec{F}_i = -\sum_{j \neq i} G \frac{m_i m_j (\vec{r}_i - \vec{r}_j)}{|\vec{r}_i - \vec{r}_j|^3} - \vec{\nabla} \cdot \phi_{ext}(\vec{r}_i),$$

when the distance between 2 objects approaches 0, equation 1 presents a singularity.
In order to avoid this, a softening length may be introduced, so the equation changes to:

$$\vec{F}_i = -\sum_{j \neq i} \frac{G m_i m_j (\vec{r}_i - \vec{r}_j)}{(|\vec{r}_i - \vec{r}_j|^2 + \epsilon^2)^{3/2}},$$

The total potential field is the sum of an external potential plus the self-consistent field defined from the distribution function itself through the solution of the poisson equation:

$$\nabla^2 \phi(\vec{x}, t) = 4\pi G \rho(\vec{r}, t),$$

where

$$\rho(\vec{r}, t) = \int f(\vec{x}, \vec{v}, t) d^3 v .$$

Numerical Approach and Testing

Using finite difference approximation

$$\frac{\vec{F}}{m} = f'(\vec{v}_0) = \frac{f(\vec{v}_0 + \Delta t) - f(\vec{v}_0)}{\Delta t}$$

The problem will be solved by using FDA to calculate forces and acceleration based on the particle's previous position and velocity.

Testing

There are two fundamental relations to check the accuracy of the solution:

The law of conservation of Energy:

$$E = \frac{1}{2}\sum_{i=1}^{N} m_i v_i^2 - \sum_{i=1}^{N}\sum_{j\neq i}^{N} G\frac{m_i m_j}{|\vec{r}_i - \vec{r}_j|}$$

The conservation of total Angular Momentum:

$$J = \sum_{i=1}^{N} \vec{r}_i \times m_i \vec{v}_i$$

Project timeline

DATE	ACTIVITY
10/13 – 10/24	Do basic research, derive equations & begin code design
10/25 – 11/15	Implement code
11/16 – 11/19	Test code
11/20 – 11/25	Run numerical experiments, analyze data, begin report
11/26 – 11/28	Finish report
12/02	Submit project

Monte Carlo Simulation of the $2D$ Ising Mode with Metropolis Algorithm

PHYS 210 Term Project Proposal

Mengxi Daisy Zhang

University of British Columbia

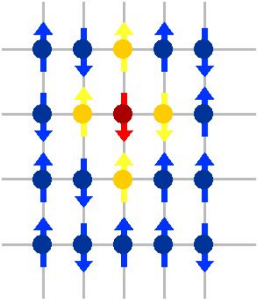
mengxiz@physics.ubc.ca

October 19, 2014

Overview

Ising Model

- Ferromagnet (e.g. iron)
- Magnetic dipole moments of atomic spins of $+1$ or -1 (up or down)
- Spin only interact with its neighbors
- Temperature and fluctuations (curie temperature)
- $2D$ is the simplest model that shows phase transition



[the Naive] Monte Carlo Method

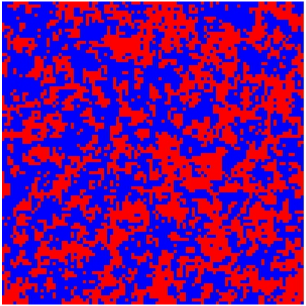
- Computational Algorithm
- Pseudo-random number
- Repeated random sampling of many states
- Compute Boltzmann factor of those random states
- Compute the corresponding thermodynamic quantities

Monte Carlo Method with Importance Sampling - The Metropolis Algorithm

- Importance sampling technique
- Choose states based on their Boltzmann factors and favor the lower ones.

Project Goals (1) - Visualizations

- Visualize the $2D$ ising model
- Generalize the code to simulate a $3D$ scenario



Project Goals (2) - Thermodynamic Quantities

- Phase transition
- Average energy
- Magnetization
- Cluster sizes
- Correlation function (dipole correlated overdistance)

More on Ising Model

Partition function is

$$Z = \sum_{s_i} e^{-\beta U} \tag{1}$$

Where U is the total energy of the system for all the interactions.

$$U = \sum_{\substack{pairs \\ i,j}} s_i s_j \tag{2}$$

Where s_i and s_j are either 1 or -1 (pointing up or down)

Monte Carlo and Metropolis Algorithm

- Start with a random state.
- Loop begins - Choose a dipole at random and consider the state where its alignment is flipped
- Compute Energy Difference using equation (2).
- If the energy U decreases, change to the state and move back to the start of the loop.
- If the energy U increases, for $e^{\Delta U/kT}$ of the chance, change to the state and then move back to the start of the loop; in other cases, directly move back to the start of the loop without changing the state.
- After many iterations, the loop ends.
- We can then compute the desired thermodynamic quantities of such a system.

Visualization and Plotting Tools

- Matlab for plots
- xvs for interactive analysis and animations

Testing

- Compare results with the known properties of the ising model. e.g. the critical points of phase transitions.
- For example, cases where the temperatures are extremely high or low.

Numerical Experiments

- Phase transition and curie temperature
- Average energy v Temperature
- Magnetization v Temperature
- Cluster sizes v Temperature
- Correlation function (dipole correlated overdistance) v Temperature

Project Timeline

Tasks	Dates
Research	Oct 17 - Oct 23
Coding	Oct 23 - Nov 10
Testing, Debugging, and Optimizing	Nov 10 - Nov 17
Numerical Experiments and Data Analysis	Nov 17 - Nov 25
Finish up Final Report	Nov 25 - Nov 31
Proofread Final Report	Dec 01 - Dec 02
Submit Final Report	Dec 03

References

Schroeder, Daniel V. An Introduction to Thermal Physics. San Francisco, CA: Addison Wesley, 2000. Print.

Thank You!

Gravitational N-Body Simulation in 2 Dimensions

PHYS210 Term Project Proposal

Mona Zhao

Overview

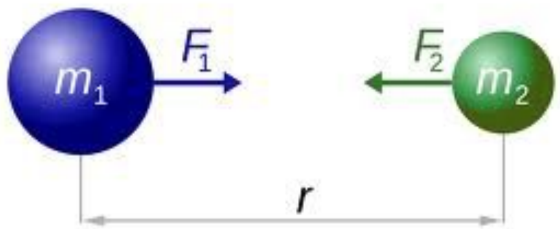
- N-body problem predicts the individual motion of n objects in a system interacting with each other gravitationally.
- To simplify the conditions, general relativity is not considered.

Goals

- Using MATLAB, write a code that simulates the effect of gravity on n number of objects over an appropriate time period.
- Experiment with a number of different initial conditions.
- Present this simulation graphically.

Mathematical Formulation

- **In 2D, Newton's law of gravity is given by:**



$$F_1 = F_2 = G\frac{m_1 \times m_2}{r^2}$$

- **Newton's Second Law of Motion:**

$$\mathbf{F} = m\frac{\mathrm{d}\mathbf{v}}{\mathrm{d}t} = m\mathbf{a}$$

Numerical Approach

- **Finite Difference Approximations (FDAs) will be used.**

$$\partial_x f = \lim_{dx \to 0} \frac{f(x+dx) - f(x)}{dx}$$

- **a = F / m = dv/dt = f(v+Δt)-f(v) / Δt**

Visualization

- MATLAB will be used for plotting.
- Animations will be generated and recorded under mpeg format.

Testing and Numerical Experiments

- The first test has simple initial conditions: 2 stationary objects with equal non-zero masses.
- If the first test passes, more complex conditions can be tested. This includes adding more objects, changing masses, and varing velocities.
- Compare results to other models if possible.
- Make sure energy and momentum are conserved.

Timeline

Date	Activity
10.13 - 10.26	Basic research, derive equations
10.27 - 11.11	Implent code
11.12 - 11.16	Test code
11.17 - 11.23	Analyze data
11.24 - 12.2	Write report
12.2	Submit project

References

- http://en.wikipedia.org/wiki/N-body_problem
- http://en.wikipedia.org/wiki/Newton%27s_law_of_universal_gravitation
- http://www.geophysik.uni-muenchen.de/~igel/Lectures/NMG/02_finite_differences.pdf