Project 2 Draft Computational Essay

Aydin O'Leary and Laurel Mitton
Prof: Jason Woodard
Class: Modeling and Simulation of the Physical World
Olin College of Engineering

28th October 2019

1 Question

If we were to build a solar house where Rae-Anne's house currently is, what's the most heat efficient configuration of materials to keep the house livable?

2 Model

Assumptions

- House is cubical (or geometrically equivalent to a cube)
- Roof is mathematically flat (see above)
- No auxiliary heating system (yet)
- Windows are double-paned (triple-paned is expensive)

Things to Implement

- Actual sun heating (Thermal mass, sun movement, realistic temperature change)
- Fix ambient temperature equation/changing
- Implement aux. heating
- Harvest Needham weather data
- Plot more data (configurations)
- Interpret graphs

```
import matplotlib.pyplot as plt

class House:
    def __init__(self):
        self.side_length       = 50
        # length of side of house (assuming house is a rectangular prism)
        self.area              = self.side_length**2
```

```
        # square footage estimate
        self.height            = 25
        # feet tall (2 story house)
        self.volume            = self.area * self.height
        # in cubic feet
        self.current_temp      = 70
        # start out with room temperature in deg F
        self.pane_surface_area = 0.08 * self.area
        # window area as a percent of square footage * area
        self.wall_surface_area = (self.side_length * self.height) - self.pane_surface_area
        self.roof_surface_area = self.area
        # assuming roof equivalent to being flat
        self.pane_insulation   = 3.4
        # efficiency of pane insulation in retaining heat (R value) (3 (2 pane) - 5(3 pane))
        self.wall_insulation   = 20
        # efficiency of wall insulation in retaining heat (R value) (19-21)
        self.roof_insulation   = 49
        # efficiency of roof insulation in retaining heat (R value) (38-60)
        # self.concrete_mass     =
        # self.water_mass        =
        self.thermal_mass      = 400
        # pounds of water

    def update(self, timestep, ambient_temp): # timestep is 1 hour or so
        # heat_transfer in BTU, surface_area in square feet, temp_a is larger temp in F,
        # temp_b is smaller temp in F, r_value is square feet * F / BTU
        pane_heat_transfer   = (self.pane_surface_area *
        abs(self.current_temp - ambient_temp))/(self.pane_insulation)
        wall_heat_transfer   = (self.wall_surface_area *
        abs(self.current_temp - ambient_temp))/(self.wall_insulation)
        roof_heat_transfer   = (self.roof_surface_area *
        abs(self.current_temp - ambient_temp))/(self.roof_insulation)
        total_heat_transfer  = -((0.30 * pane_heat_transfer) + (0.40 * wall_heat_transfer)
        + (0.30 * roof_heat_transfer))
        # maybe add floors in future (accounts for 15% but for now all others are up by 5%)
        self.current_temp    += total_heat_transfer / self.thermal_mass
        return self.current_temp

class World:
    def __init__(self):
        self.cloudy       = False # is it cloudy today? true/false
        self.sun_angle    = 45
        # angle against the movement of the sun (altitude from horizon)
        # between 24 deg (jan) and 71 deg (jun)
        self.sun_position = 90
        # angle with the movement of the sun (from east to west)
        # between 0 deg (sunrise), 180 deg (sunset), 359 deg (before sunrise)
        self.ambient_temp = 60
```

```
        # deg. fahrenheit - between 30 deg (jan) and 74 deg (july)

    # various update functions so the world can change over the course of a day
    def flipCloudy(self):
        self.cloudy = (not self.cloudy)
    def changeSunAngle(self, new_angle):
        self.sun_angle = new_angle
    def changeSunPosition(self, new_position):
        self.sun_position = new_position
    def changeAmbientTemp(self, new_temp):
        self.ambient_temp = new_temp

if __name__ == '__main__':
    # running model once = simulating one day
    house = House()
    world = World()
    timestep = 1
    time_log = []
    house_temp_log = []
    world_temp_log = []
    sim_length = 20 # hours
    for i in range(sim_length):
        house_current_temp = house.update(timestep,world.ambient_temp)
        # update house temperature
        time_log.append(i)
        house_temp_log.append(house_current_temp)
        world_temp_log.append(world.ambient_temp)
        print(house_current_temp)
        world.changeAmbientTemp(world.ambient_temp - i)
        # progressively getting colder
        # note: this is a very inaccurate range for world temperature, as a stand in for now
    # graphing data
    plt.figure(1)
    plt.plot(time_log, house_temp_log, '.', label='House Temperature')
    plt.plot(time_log, world_temp_log, '.', label='World Temperature')
    plt.xlabel('Time (hrs)')
    plt.ylabel('Degrees (F)')
    plt.legend()
    plt.show()
```

3 Results

Currently, we are logging and graphing the house and world temperatures. The graph shows how the house temperature responds to changes in the ambient temperature, and we plan to expand our graphing to show various configurations of house and compare the results. In our "default" version, the house temperature falls as the world temperature does, though not as steeply.

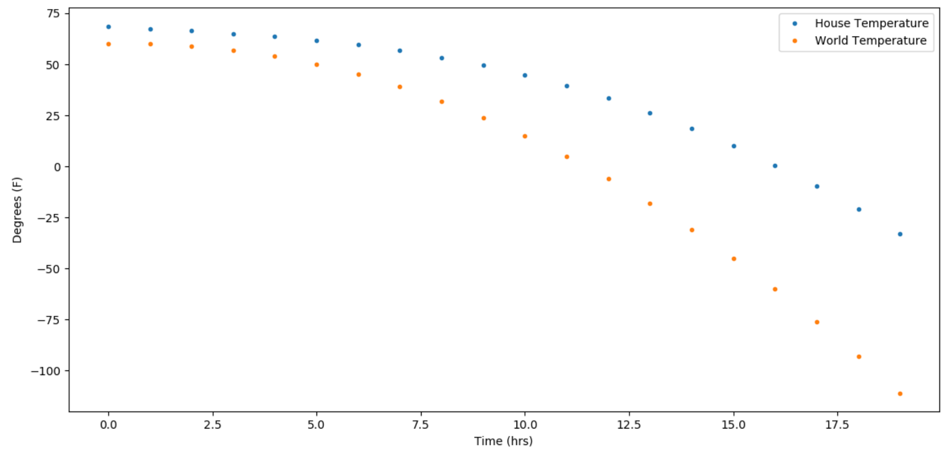

Figure 1: Sample results

4 Interpretation

We would be able to determine optimal number and size of windows, necessity and (if needed) efficiency of auxiliary heating, and other characteristics of the house. An ideal house would stay between 55 and 85 F (range taken from dorm thermostat).