Introduction to the Finite Difference Method: Filling and Draining a Cylindrical Tube

Lensyl Urbano

Wednesday 2nd November, 2022

Chapter 1

Filling a Cylindrical Tube

Consider filling a cylinder with water.

The water flows in at a constant rate of 5 cm^3/s. The inflow rate (Q) can be written as the change in volume (V) over the change in time (t) (the Δ symbol represents change):

Inflow Rate

$$Q = \frac{\Delta V}{\Delta t} = 5\ cm^3/s \tag{1.1}$$

1.1 Filling the Tube Calculations and Equations

1.1.1 Conceptual Physics Approach

So, at this inflow rate, after 10 seconds there will be 50 cm^3 added to the cylinder.

$$\begin{aligned} V &= 5\ cm^3/s \cdot 10\ s \\ &= 50\ cm^3 \end{aligned}$$

In terms of the equation, the change in volume is equal to the inflow rate (Q) times the time period (t).

$$V = Q \cdot t \tag{1.2}$$

How high will the water have risen in the cylinder in those 10 seconds when 50 cm^3 of water was added? Well, we know that the

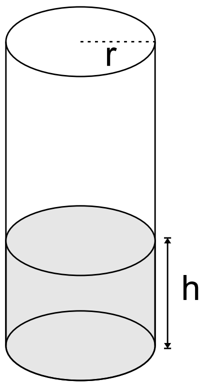

Figure 1.1: Cylinder with dimensions. r is the radius and h is the height of water in the tube.

volume of a cylinder is given by the equation:

$$V = \pi r^2 h \tag{1.3}$$

So, if we know the volume and the radius of the cylinder (r) we can solve this equation for the height (h):

Divide both sides by πr^2:

$$\frac{V}{\pi r^2} = \frac{\pi r^2 h}{\pi r^2} \tag{1.4}$$

To get:

$$\frac{V}{\pi r^2} = h \tag{1.5}$$

Which can be rewritten as:

$$h = \frac{V}{\pi r^2} \tag{1.6}$$

or:

$$h = \frac{1}{\pi r^2} V \tag{1.7}$$

Thus, for our given problem where the radius is 2.25 cm, and the volume of water added is 50 cm^3:

$$h = \frac{1}{\pi \cdot 2.25^2} \cdot 50 = 3.1 \text{ cm} \tag{1.8}$$

Now we can substitute for volume using Equation 1.2 to get:

$$h = \frac{1}{\pi r^2}\, Q \cdot t \tag{1.9}$$

Now, lets rewrite this equation so we just consider what happens over a small time period (call it a *time step* denoted by Δt). It's the change from moment to moment and results in a small change in height (Δh). So our final equation becomes:

$$\Delta h = \frac{1}{\pi r^2}\, Q \cdot \Delta t \tag{1.10}$$

Which we rearrange a little to get:

$$\boxed{\Delta h = \frac{\Delta t}{\pi r^2}\, Q} \tag{1.11}$$

Having calculated the change in the height of the water in the cylinder in a given time step (Δt), for each timestep we calculate the new height of water (h_{new}) as the old height plus the change:

$$\boxed{h_{new} = h_{old} + \Delta h} \tag{1.12}$$

We can now use these two equation to create a computer model that gives the height of water in the column over time.

1.1.2 Using Calculus to Find the Discrete Equations

Same problem–filling a cylinder–but using calculus to end up with the same equations in the end.

Start with the equation for the volume of a cylinder:

$$V = \pi r^2 h \tag{1.13}$$

There are two variables that change with time as the cylinder fills, the volume (V) and the height (h) since the radius (r) does not change. So, if we differentiate this equation with respect to time (implicit differentiation), we get:

$$\frac{dV}{dt} = \pi r^2 \frac{dh}{dt} \tag{1.14}$$

Solving for $\frac{dh}{dt}$ gives the **height change equation**:

$$\frac{dh}{dt} = \frac{1}{\pi r^2}\frac{dV}{dt} \tag{1.15}$$

The expression $\frac{dh}{dt}$ represents the instantaneous change in height with time: the rate at which height changes at any instant. To write a program to solve this equation we'll **discretize** the expression by using $\frac{\Delta h}{\Delta t}$:

$$\frac{\Delta h}{\Delta t} \approx \frac{dh}{dt} \tag{1.16}$$

The Δ means that we're taking the difference between two discrete value of h, so:

$$\Delta h = h_2 - h_1 \tag{1.17}$$

Since this is the rate of change over time it can be easier to think of the change in height as the difference between the new height and the old height over the short (Δt) time period.

$$\Delta h = h_{new} - h_{old} \tag{1.18}$$

So now we rewrite our height change equation (Eqn. 1.15) as:

$$\frac{\Delta h}{\Delta t} = \frac{1}{\pi r^2} \frac{dV}{dt} \tag{1.19}$$

which we can solve for the change in height (Δh):

$$\Delta h = \frac{\Delta t}{\pi r^2} \frac{dV}{dt} \tag{1.20}$$

Since the inflow rate (Q) is the change in volume over time, and it remains constant for our model, we can say:

Change in Height Equation

$$\boxed{\Delta h = \frac{\Delta t}{\pi r^2} \, Q} \tag{1.21}$$

Which is the same equation (Eqn. 1.11) we found when we took the conceptual approach in the previous section.

Now we substitute in the discrete change for Δh (Eqn. 1.18) to get:

$$h_{new} - h_{old} = \frac{\Delta t}{\pi r^2} \, Q \tag{1.22}$$

Which we can solve for the new height:

$$h_{new} = h_{old} + \frac{\Delta t}{\pi r^2} \, Q \tag{1.23}$$

Which is the same as:

Height Update Equation

$$\boxed{h_{new} = h_{old} + \Delta h} \tag{1.24}$$

We can use the **Change in Height** (Eqn. 1.21) and **Height Update** (Eqn. 1.18) equations to create a computer model of the height of the water in the cylinder as it fills it up.

1.2 Code

The following example code that solves this water-filling problem uses the ezGraph class (https://github.com/lurbano/ezGraph) which requires matplotlib and numpy. However, the code in the following section avoids the use of most imported modules, but does not graph.

Code with Graphical Output

water-filling-fd.py

```python
import numpy as np
import time
from ezGraph import *

# Finite Difference Model

# PARAMETERS
dt = 1.
nsteps = 20

r = 2.25      # radius (cm)
Qin = 5       # Volume inflow rate: (cubic cm / s)
h = 0         # Initial height (cm)

# GRAPH
graph = ezGraph(xmax=30, ymin=0, ymax=10, xLabel="Time (s)", yLabel="Height (cm)")
graph.add(0, h)                    # add initial values


# TIME LOOP
for t in range(1, nsteps):
    modelTime = t * dt

    dh = Qin * dt / (np.pi * r**2)     # find the change in height
    h = h + dh                         # update height

    print(modelTime, h)
    graph.add(modelTime, h)
```

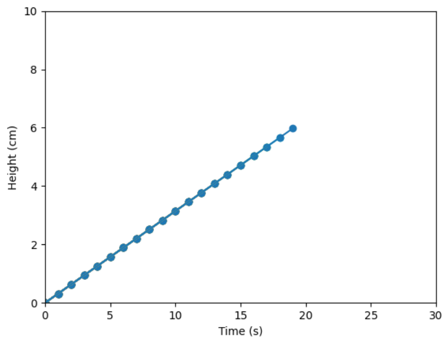

Figure 1.2: Model output: Graph of height of water in the column over time when filling the cylinder.

```
    graph.wait(0.1)

# DRAW GRAPH
graph.keepOpen()
```

1.2.1 Code without Graphical Output

A stripped down version of the code with no graph and no external modules except ”math”.

water-filling-fd-noGraph.py

```
import math
# Finite Difference Model

# PARAMETERS
dt = 1.
nsteps = 20

r = 2.25        # radius (cm)
Q = 5           # Volume inflow rate: (cubic cm / s
    )
h = 0           # Initial height (cm)

print(0, h)       # print initial values

# TIME LOOP
for t in range(1, nsteps):
    modelTime = t * dt

    dh = Q * dt / (math.pi * r**2)      # find
        the change in height
    h = h + dh                          # update
        height

    print(modelTime, h)
```

Which should produce a table of time and height output:

```
0 0
1.0 0.31438013450250935
2.0 0.6287602690050187
3.0 0.943140403507528
4.0 1.2575205380100374
5.0 1.5719006725125468
```

```
6.0 1.8862808070150563
7.0 2.2006609415175657
8.0 2.515041076020075
9.0 2.8294212105225847
10.0 3.143801345025094
11.0 3.4581814795276036
12.0 3.772561614030113
13.0 4.086941748532622
14.0 4.4013218830351315
15.0 4.715702017537641
16.0 5.03008215204015
17.0 5.34446228654266
18.0 5.658842421045169
19.0 5.973222555547679
```

1.3 Analytical Solutions using Calculus

The analytical solution to this problem will help confirm the accuracy of our model.+

1.3.1 Filling

As we saw in the section on using calculus (Section 1.1.2), we can start with the equation for the volume of a cylinder:

$$V = \pi r^2 h \tag{1.25}$$

And differentiate with respect to time to get:

$$\frac{dV}{dt} = \pi r^2 \frac{dh}{dt} \tag{1.26}$$

Assuming a constant inflow rate ($\frac{dV}{dt} = Q$):

$$Q = \pi r^2 \frac{dh}{dt} \tag{1.27}$$

And solve for $\frac{dh}{dt}$:

$$\frac{dh}{dt} = \frac{Q}{\pi r^2} \tag{1.28}$$

This we can separate:

$$dh = \frac{Q}{\pi r^2} dt \tag{1.29}$$

and integrate:

$$\int dh = \int \frac{Q}{\pi r^2} dt \tag{1.30}$$

to get:

$$h = \frac{Q}{\pi r^2} t + c \tag{1.31}$$

When $t = 0$, c can be shown to be the initial height ($c = h_i$) so:

$$h = \frac{Q}{\pi r^2} t + h_i \tag{1.32}$$

And since $\frac{Q}{\pi r^2}$ is constant, we can see that this term is the slope in a linear equation of the form.

$$y = mx + b \tag{1.33}$$

So the linear pattern produced by the filling model is correct (Figure 1.2).

Chapter 2

Draining

Consider a cylinder with water draining out of the bottom through a hole.

2.1 Numerical Solution

For draining, the outflow rate (change in volume over time, ($Q = \frac{dV}{dt}$) is not constant. The outflow rate is proportional to the height of water in the tube, since the higher the water level the greater pressure at the bottom of the tube and the faster the outflow rate.

$$\frac{dV}{dt} \propto h \tag{2.1}$$

Converting the proportionality statement to an equation requires us to introduce a constant (k). Also, recognizing that this will be an outflow rate means that the flow rate should be negative:

$$\frac{dV}{dt} = -k \cdot h \tag{2.2}$$

As we saw when we were filling the cylinder, the change in height of water in the tube is given by the **change in height** equation:

$$\boxed{\Delta h = \frac{\Delta t}{\pi r^2}\, Q} \tag{2.3}$$

where Q is $\frac{dV}{dt}$ so:

$$\Delta h = \frac{\Delta t}{\pi r^2}\, \frac{dV}{dt} \tag{2.4}$$

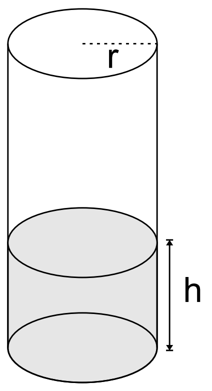

Figure 2.1: Cylinder with dimensions. r is the radius and h is the height of water in the tube.

So, let's substitute our flow rate equation (Eqn. 2.2) for $\frac{dV}{dt}$ to get:

$$\Delta h = \frac{\Delta t}{\pi r^2} \left(-k \cdot h\right) \tag{2.5}$$

which simplifies to:

$$\Delta h = -\frac{\Delta t}{\pi r^2} \, k \cdot h \tag{2.6}$$

Important to note for the computer model, is that the height (h) used in this equation is the old height from the previous timestep so:

$$\boxed{\Delta h = -\frac{\Delta t}{\pi r^2} \, k \cdot h_{old}} \tag{2.7}$$

and we can still update the new height using:

$$\boxed{h_{new} = h_{old} + \Delta h} \tag{2.8}$$

so, in our code we just need to change this line and set up a few different constants.

2.1.1 Code: Draining

This program is based off the filling code, but for this example we ignore the filling by setting the inflow rate to zero (**Line 12**). We're using an initial height of 50 cm ($h_0 = 50$), and set the constant k to be equal to one.

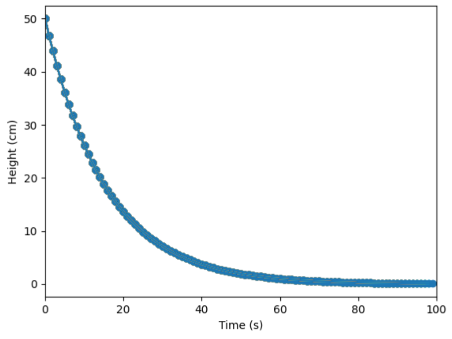

Figure 2.2: Model output: Graph of height of water in the column over time when draining the cylinder via a hole in the bottom.

water-draining-fd.py

```
1  import numpy as np
2  import time
3  from ezGraph import *
4
5  # Finite Difference Model
6
7  # PARAMETERS
8  dt = 1
9  nsteps = 100
10
11 r = 2.25     # radius (cm)
12 Qin = 5        # Volume inflow rate (dV/dt): (
       cubic cm / s)
```

```
h = 0           # Initial height (cm)
k = 1.0         # outflow rate constant

# GRAPH
graph = ezGraph(xmax=100,
                xLabel="Time (s)", yLabel="Height (cm)")
graph.add(0, h)                  # add initial values


# TIME LOOP
for t in range(1, nsteps):
    modelTime = t * dt

    # Filling
    dh = Qin * dt / (np.pi * r**2)  # find the change in height
    h = h + dh                       # update height

    # Draining
    dVdt = -k * h
    dh = dVdt * dt / (np.pi * r**2)
    h = h + dh

    print(modelTime, h)
    graph.add(modelTime, h)
    graph.wait(0.1)

# DRAW GRAPH
graph.keepOpen()
```

The output from the model (Fig. 4.1) looks like an exponential decay curve, which is what we will find from the analytical solution (Eqn. 2.24).

2.2 Draining: Analytical Solution using Calculus

Experiments (which you may have done) show that if you're draining a cylinder by gravity the outflow rate of water is linearly proportional to the height of water in the tube.

$$\frac{dV}{dt} \propto h \tag{2.9}$$

Converting the proportionality statement to an equation requires us to introduce a constant (k):

$$\frac{dV}{dt} = kh \tag{2.10}$$

So in draining, the outflow rate ($\frac{dV}{dt}$) is not constant, it slows down as the height of water in the tube decreases.

Now, lets substitute the equation for the volume of a cylinder:

$$V = \pi r^2 h \tag{2.11}$$

into the draining equation (Eq. 2.10) to get:

$$\frac{d[\pi r^2 h]}{dt} = kh \tag{2.12}$$

we can extract π and r^2 from the differential because they are constant:

$$\pi r^2 \frac{dh}{dt} = kh \tag{2.13}$$

separating the variables gives:

$$\pi r^2 \frac{dh}{h} = k \cdot dt \tag{2.14}$$

and rearranging:

$$\frac{dh}{h} = \frac{k \cdot dt}{\pi r^2} \tag{2.15}$$

$$\frac{dh}{h} = \frac{k}{\pi r^2} dt \tag{2.16}$$

To simplify a little, lets consolidate the constants on the left hand side into one variable K:

$$K = \frac{k}{\pi r^2} \tag{2.17}$$

so:

$$\frac{dh}{h} = K \cdot dt \tag{2.18}$$

which we can integrate (remember K is a constant):

$$\int \frac{dh}{h} = K \int dt \tag{2.19}$$

$$\ln h = K \cdot t + c \tag{2.20}$$

we can solve for h by raising both sides by e to cancel the ln:

$$e^{\ln h} = e^{Kt+c} \tag{2.21}$$

$$h = e^{Kt+c} \tag{2.22}$$

Because of math, we can pull the constant out to get:

$$h = Ce^{Kt} \tag{2.23}$$

Where the constant is the initial value of the height (h_0):

$$h = h_0 \cdot e^{Kt} \tag{2.24}$$

This is an exponential function. If K is less than 1 ($K < 1$) then this is a decay curve.

Chapter 3

Model Calibration

We have a model that shows the general patterns we expect when filling and draining the tube: linear increase for filling at a constant rate, and exponential decay for draining due to gravity. But can these models reflect the actual thing?

Fortunately, we have some experimental data, thanks to my pre-Calculus class.

We'll start with the filling portion of the model.

3.1 Filling Calibration

time (s)	height (cm)
1	0
7	10
12	20
16.8	30
21.5	40
26.2	50

Table 3.1: Experimental Data from filling the cylinder. Data by Blas.

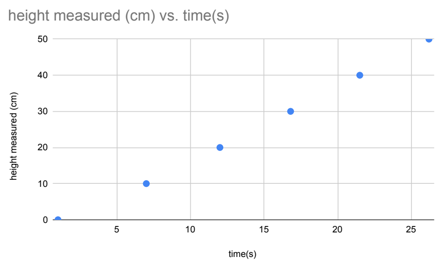

Figure 3.1: Experimental data from filling the cylinder. Data by Blas.

Recall that, at the core of the model, is the change in height equation (Eqn. 1.11):

$$\Delta h = \frac{\Delta t}{\pi r^2} Q$$

The radius (r) is measured and we choose Δt as steps in the simulation, so the only unknown is the inflow rate (Q).

Therefore, to calibrate the model we adjust Q until the model output matches the experimental data.

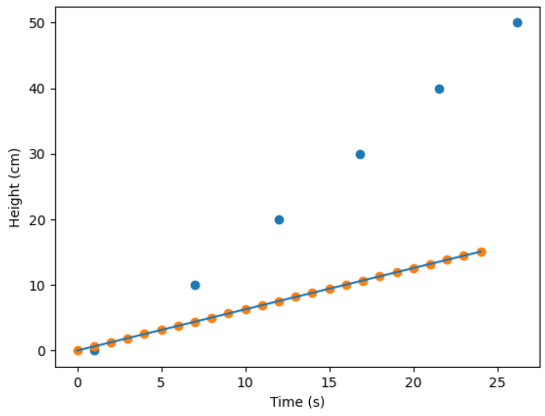

Figure 3.2: Comparison of measured (blue) and modeled (red) data. This version of the model uses $Q = 10$ cm^3/s.

In order to produce this graph, we use the *ezGraphMM* class, which allows us to plot the measured and modeled data separately. The full code is here:

water-filling-calibration.py

```
import numpy as np
import time
from ezGraph import *

def myAvg(lst):
    # sum
    s = 0
    n = 0
    for i in lst:
        s = s + i
        n += 1
    return s/n

# Finite Difference Model

# PARAMETERS
dt = 1
```

```
nsteps = 30

r = 2.25      # radius (cm)
Qin = 30        # Volume inflow rate (dV/dt): (
    cubic cm / s)
h = 0         # Initial height (cm)
k = 0.0       # outflow rate constant

# EXPERIMENTAL DATA
x_measured = [1, 7, 12, 17, 22, 26]
y_measured = [0, 10, 20, 30, 40, 50]
y_modeled = []

# GRAPH
graph = ezGraphMM(xmin=0, xmax=100,
                  xLabel="Time (s)",
                  yLabel="Height (cm)",
                  x_measured = x_measured,
                  y_measured = y_measured)

graph.addModeled(0, h)                # add
    initial values


# TIME LOOP
for t in range(1, nsteps):
    modelTime = t * dt

    # Filling
    dh = Qin * dt / (np.pi * r**2)  # find the
        change in height
    h = h + dh                        # update
        height

    # Draining
    dVdt = -k * h
    dh = dVdt * dt / (np.pi * r**2)
    h = h + dh

    if (modelTime in x_measured):
        print(modelTime, h)
        y_modeled.append(h)

    graph.addModeled(modelTime, h)
    graph.wait(0.01)

```

```
print("h_measured:", y_measured)
print("h_modeled:", y_modeled)

print(f'avg_measured = {myAvg(y_measured)}')
# calculate average values for y_measured and
    y_modeled

# DRAW GRAPH
graph.keepOpen()
```

3.2 How Good are Our Results: Finding the r^2 Value

To determine how good a match we have (instead of just eyeballing it) we'll calculate the regression coefficient (r^2) value using a series of assignments.

3.2.1 Selecting time data

Assignment 1:
Adapt the model so it records the water heights (in an array) at the same times as the measured data (you may round the measured times to whole integers to make it easier).

3.2.2 Functions: Average/Mean

The average or mean, as we all recall, is the sum of all the values divided by the number of values (n). Equation-wise, if we have a set of height (h) values it looks like:

$$\text{average} = \frac{\sum h}{n} \tag{3.1}$$

Assignment 2:
Write a function that can find the average value of all the values in an array. There is a built in average function, but don't use it because you'll need to practice going through loops and finding sums for the rest of these assignments.

3.2.3 Sum of the errors/residuals

The *error* is simply the difference between the measured and the modeled value. The error is sometimes referred to as the **residual**.

$$e = h_{measured} - h_{modeled} \tag{3.2}$$

The sum of all the errors (E) can give an idea of how well you model matches the actual data.

$$E = \sum e = \sum (h_{measured} - h_{modeled}) \tag{3.3}$$

Assignment 3:
Write a function that takes the measured heights and the corresponding modeled heights and returns the **sum of all the errors** (e).

3.2.4 Absolute Error

As you may have noticed, the sum of the errors is not the best measure of how good your model fits the measured data because, if you have two errors in opposite directions, they can cancel each other out, resulting in a low error value.

To account for this problem, you can get the sum of the absolute values of the errors (S_{abs}).

$$S_{abs} = \sum |h_{measured} - h_{modeled}| \tag{3.4}$$

Assignment 4:
Write a function that finds the sum of the absolute values of the errors.

3.2.5 Sum of the errors squared

Another alternative to using the absolute value is to square the error instead. This achieves the same goal of making all the errors positive so the sum of the error squared (S_{sq}) is a decent metric of the model's fit.

$$S_{sq} = \sum (h_{measured} - h_{modeled})^2 \tag{3.5}$$

Assignment 5:
Write a function that finds the sum of the square of the errors.
Adjust the parameters in your model to get the lowest S_{sq} value you can.

3.2.6 Differences from the mean

We adjust the parameters in our model to minimize the sum of the errors squared (S_{sq}) to find the best fit, we are calibrating the model. But while this is a good way of calibrating this specific model, it's does not allow us to see how well the model matches the data in the broader scheme of things: How does this model of water draining compare to other models where the tank is a different size or shape for example.

To get around this we can try to *nomralize* S_{sq}. The convention is to divide it by the differences of the measured values from their mean ($/mu$) squared:

Where:

$$\begin{aligned} \mu &= \text{average of the measured data} && (3.6) \\ &= \frac{\sum h_{measured}}{n} && (3.7) \end{aligned}$$

Therefore the sum of the differences squared is:

$$D_{sq} = \sum (h_{measured} - \mu)^2 \tag{3.8}$$

You already have a function that finds the mean/average, now:

Assignment 6:
Write a function that finds the sum of all the differences between the **measured** heights and their mean).

3.2.7 r^2

Finally, we can find the regression coefficient. A regression coefficient of 1 is a perfect fit.

Assignment 7:
The regression coefficient (r^2) is given by one minus the sum of all the errors squared divided by the sum of the differences from the mean squared:

$$r^2 = 1 - \frac{S_{sq}}{D_{sq}} \tag{3.9}$$

Write a function that calculates the r^2 value.

3.3 Draining Calibration

time (s)	height (cm)
0	50
28	40
60	30
101	20
157	10

Table 3.2: Experimental Data from draining the cylinder. Data by Coen and Blas.

Assignment 8:
Use this experimental data to calibrate the draining portion of the model. Now you're trying to get the regression coefficient as close to 1 as possible.

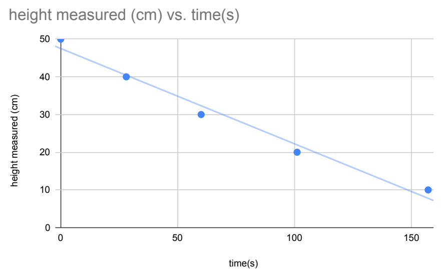

Figure 3.3: Experimental data from draining the cylinder. A linear trendline is included to highlight the curvature of the data. Data produced by Coen and Blas.

3.4 Breaking the Model

Experiment by changing the parameters to find the limits of the model. Questions to ask are:

1. What is the largest timestep you can use with this model.
2. What happens when you change the k coefficient? How large can you make it?

What you should find is that at for some values the model starts to oscillate out of control. This is one of the limitations of finite difference models. It requires pretty extreme settings in this example, but other models the unstable parameter values can be much closer to the values we want to use.

Chapter 4

Expanding the Model

As we've seen, modifying the basic model for different conditions can be pretty easy. In the base model (water filling) the change in height equation (Eqn. 1.11) was:

$$\Delta h = \frac{\Delta t}{\pi r^2}\, Q$$

To adjust for draining (Eqn. 2.7), we just needed to make the inflow value (Q) an equation that adjusts for the fact that the water outflow rate is negative and depends on the height of water in the tube.

$$\Delta h = \frac{\Delta t}{\pi r^2}\,(-k \cdot h)$$

4.1 Simultaneously Filling and Draining

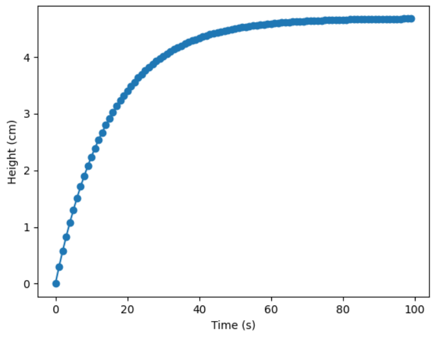

Figure 4.1: The model indicates that filling the cylinder while the hole is still open would result in the water rising to an equilibrium height of about 4.7 cm.

Our program is already set up to handle filling and draining. We simply need to change the inflow value (Q_{in}).

What happens if we try to fill the tube while the hole in the bottom is open so it simultaneously drains?

Set the inflow value to 5 cm^3/s, set the initial height (h) to 0, and keep the draining parameters, we should see the water level increase until it reaches an equilibrium.

Model Equilibrium: Now, if you have not already, set the inflow value and k coefficient to the values you got from your calibrations and determine the equilibrium height of water in the tube.

4.2 Time-variable Inputs: Changing Inflow

4.2.1 Stopping Inflow

Assignment 9:
Adapt the model so that the water input stops after 50 seconds. Use the calibrated fill rate (Q_{in}) for the initial inflow rate, and allow the model to drain using the calibrated k coefficient.

4.2.2 Variable Inflow

Assignment 10:
Change the model so that the water input varies in a sinusoidal (wave) pattern with the function:

$$Q = 10 \sin(5t) + 11 \tag{4.1}$$

Allow the model to drain using the calibrated k value.

4.2.3 DYO Simulation

Assignment 11:
Design your own simulation scenario and share it with the class.

4.3 Water Tank Exercises

4.3.1 Rectangular Prism

Assignment 12:
Create a model for the filling of a water tank that is a rectangular prism with a width of 10 cm and a length of 5 cm.

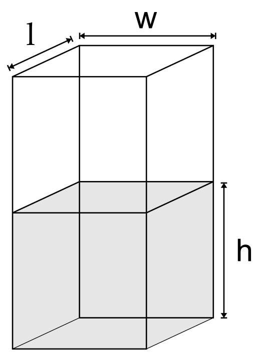

Figure 4.2: Water Tank

4.3.2 Cone

Assignment 13:
Create a model for the filling of a water container where the container was conical instead of a cylinder, where the radius increases by 1 cm for every 2 cm increase in height?

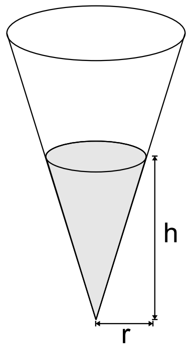

Figure 4.3: Conical water container.

Chapter 5

Ballistic Motion: A ball in the Air

Here we'll see how to create models based off of governing differential equations. First we attempt to find the height of a ball that is thrown upwards.

5.1 Acceleration and Velocity

For a ball flying through the air, the only force acting on it is gravity (we're ignoring friction with the air for now). If we do the physics we find that the acceleration (a) of a ballistic object is constant no matter the size or mass of the object.

As we know, acceleration is the change in velocity with time so:

$$\boxed{a = \frac{dv}{dt}} \tag{5.1}$$

This we can **discretize** as such:

$$\frac{dv}{dt} = \frac{\Delta v}{\Delta t} \tag{5.2}$$

so that:

$$a = \frac{\Delta v}{\Delta t} \tag{5.3}$$

since velocity is changing over time:

$$a = \frac{v_{new} - v_{old}}{\Delta t} \tag{5.4}$$

So if we know our starting velocity (v_{old}) and our acceleration and set a timestep (Δt) we can solve this equation to calculate the new velocity:

$$\boxed{v_{new} = v_{old} + a \cdot \Delta t} \tag{5.5}$$

5.2 Height and Velocity

What we want to find is the height of the ball, and we do so by recalling that velocity (in an up-down direction) is the change in height (y) over time:

$$v = \frac{dy}{dt} \tag{5.6}$$

Which we can discritize in the same way we did for acceleration to get a very similar equation to Eqn. 5.5 to calculate height (y):

$$\boxed{y_{new} = y_{old} + v \cdot \Delta t} \tag{5.7}$$

With these two equations (Eqns. 5.5 and 5.7), we can now create a model to simulate the height of the ball.

5.3 Model

5.3.1 Parameters

First we set our **parameters**. Acceleration due to gravity (g) is:

$$g = -9.8\ m/s^2 \tag{5.8}$$

It's negative because it's acting downwards.

We also need to set a timestep ($\Delta t = 0.1$)and a length for the simulation, which we set as the number of steps (45 steps).

5.3.2 Initial Values

Next we set the initial values:

$$v_0 \quad = 20\ m/s \tag{5.9}$$
$$y_0 \quad = 0\ m \tag{5.10}$$

We will also need to initialize our output, which in this case is the graph.

5.3.3 Time Loop

Run through the given number of timesteps and calculate the height in each iteration.

5.3.4 Code

The final code may look like this:

```
import numpy as np
import time
from ezGraph import *

# Finite Difference Model of the
#  Height of a Ballistic Ball

# PARAMETERS
dt = 0.1
nsteps = 45

g = -9.8      # acceleration due to gravity (m/s
    ^2)

# INITIAL VALUES
h = 0         # initial height (m)
v = 20        # initial velocity (m/s)

# GRAPH
graph = ezGraph(xmin=0, xmax=100,
                xLabel="Time (s)",
                yLabel="Ball Height (m)")

graph.add(0, h)      # add initial values

# TIME LOOP
for t in range(1, nsteps):
    modelTime = t * dt

    # First find the velocity
    v = v + g * dt

    # Find new height
    h = h + v * dt

    graph.add(modelTime, h)
    graph.wait(0.1)

# DRAW GRAPH
graph.keepOpen()
```

5.4 Analysis

5.4.1 Maximum Height

Assignment 14:
Have your program find the maximum height of the ball and the time when it reaches it. Compare your result to the analytical solutions (look them up).

5.4.2 Hitting the Ground

Assignment 15:
Have your program find the moment when the ball hits the ground.

Chapter 6

Energy Balance Model

If there were no atmosphere, what would the average temperature at the surface of the Earth?