Bottle Pitches

Axel Tong (3H1), Tey Yi Fan, Chow Guan Ze (3S1)

Project Work 2019
Group 8–06

1 Introduction

There is an ideal cylindrical container of base radius R, border thickness $d > 0$, and infinite height. A cylindrical laminar stream of water (with radius ϵ) is shot into the container, impacting the centre of the circular base at constant velocity v. This produces an audible sound, consisting of different amplitudes of waves of different frequencies. The weighted average of these amplitudes is said to be the *pitch* of the sound, which we denote by S.

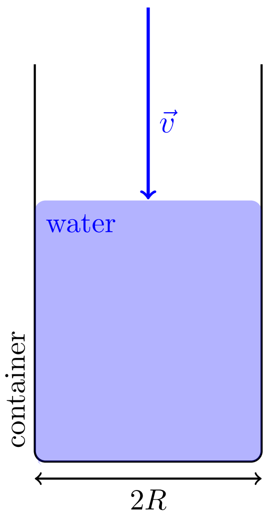


Figure 1: A cross-sectional diagram of our system.

The problem at hand is to quantitatively describe S, depending on the parameters v, R and d. We treat $\epsilon > 0$ as a constant small real number.

The mathematical model we will construct will be based on physical theory of how the system should behave. It predicts that the impact of the stream of water causes vibrations in the circular base, which can be expressed as a sum of sinusoids at different frequencies. Each frequency exactly corresponds to a pitch of sound. Furthermore, no contribution to the pitch of is made due to vibrations of the side walls. (They only contribute towards the magnitude of the sound.)

Using various methods explained in the following section, we will break up the discussion of our problem into three steps.

1. Using *the method of finite differences*, find a Partial Differential Equation (PDE) which describes the vibration of the circular base.
2. Using numerical methods, obtain a precise power series approximation to the solution of the PDE.
3. Using the Fourier transform, find the approximate Fourier coefficients, then find their weighted average to find S.

Before proceeding to solve the problem, we introduce some preliminaries necessary.

2 Preliminaries

2.1 Partial Differential Equations and Finite Difference Methods

Let $f : \mathbb{R}^n \to \mathbb{R}$ be infinitely differentiable in each of the n variables $x_1, x_2, \ldots, x_n$. A *Partial Differential Equation*, or PDE, in f is an equation involving f and all its partial derivatives in the n variables. To *solve* a PDE is to find all functions f so that the equation is satisfied for all $(x_1, x_2, \ldots, x_n) \in \mathbb{R}^n$.

The *del operator* is the vector operator $\nabla = (\partial_1, \partial_2, \ldots, \partial_n)$. If $F : \mathbb{R}^n \to \mathbb{R}$ is a scalar field, then $\nabla F = (F_1, F_2, \ldots, F_n)$ is called the *gradient field* of F, and measures how quickly F is growing at any particular point in $\mathbb{R}^n$. If $F : \mathbb{R}^n \to \mathbb{R}^n, (x_1, x_2, \ldots, x_n) \mapsto (y_1, y_2, \ldots, y_n)$ is a vector field, then $\nabla \cdot F = \partial_1 y_1 + \partial_2 y_2 + \cdots + \partial_n y_n$ is called the *divergence* of F. If $F : \mathbb{R}^n \to \mathbb{R}$ is a scalar field, the quantity $\nabla^2 F = \nabla \cdot (\nabla F)$ is called the *Laplacian* of F, and the operator ∇^2 is called the *Laplacian operator*, or simply the *Laplacian*.

Imagine a 2-dimensional plane in a 3-dimensional space. Imagine further that each point on the plane can at any time obtain a certain height above or below zero, and that this height is given by the function $F(x, y)$, where (x, y) is the coordinate of the point. In this case, the Laplacian $\nabla^2 F(x, y)$ describes, on average, *the negative of* how much higher the point (x, y) is than its neighbours. In particular, if $F(x, y)$ is on average much greater than $F(x', y')$ for $\sqrt{(x - x')^2 + (y - y')^2} \ll 1$, then the Laplacian $\nabla^2 F(x, y)$ would be very *small* (negative), and vice versa. The Laplacian is in this sense a generalisation of the second derivative of single variable calculus to arbitrary scalar fields in n variables, in a way that takes into account contributions of all variables.

The theory of partial differential equations is the problem of determining $F(x_1, x_2, \dots, x_n)$ at all points $(x_1, \dots, x_n) \in \mathbb{R}^n$. Clearly, $\mathbb{R}$ is a continuum, so this problem is an infinite one. In many cases, it is easier to consider similar notions of PDEs over finite spaces, where one or more x_i only take on discrete values. If we are not solving F over the whole real line and one of the x_i only ranges within an interval $[a, b]$, then a common way to perform the discretisation is to divide the interval into k pieces, and let x_i take the end values. That is, we only consider the discrete values $x_i = a, a + (b - a)/k, a + 2(b - a)/k, \dots, a + k(b - a)/k$. When we let $k \to \infty$, this discrete approximation to our PDE becomes exact.

This method of making the continuous PDE discrete is known as *the method of finite differences*, or a *finite difference method.* It is mainly helpful in two ways. First, it helps when we are formulating the PDE to be solved, because it is much easier to consider forces (in accordance with Newtonian physics) on discrete particles than a continuum. Secondly, it helps in numerically solving PDEs when no analytical solutions exist or when they are very complicated. By letting k be a large but finite integer, a computer can carry out the finitely many computations involved in getting an approximate solution to the PDE with relative ease. Both will be used in our project.

2.2 The Fourier Series and Fourier Transform

Suppose $f : [0, P] \to \mathbb{R}$ is "sufficiently nice"[1]. Let

$$f_n(t) := \frac{a_0}{2} + \sum_{k=1}^{n} \left(a_k \cos\left(\frac{2\pi k t}{P}\right) + b_k \sin\left(\frac{2\pi k t}{P}\right) \right), \tag{1}$$

where the coefficients a_k, b_k are defined by

$$a_k := \frac{2}{P} \int_0^P f(t) \cos\left(\frac{2\pi k t}{P}\right) dt \quad \text{and} \quad b_k := \frac{2}{P} \int_0^P f(t) \sin\left(\frac{2\pi k t}{P}\right) dt.$$

Then, for all $t \in [0, P]$, we have that $\lim_{n\to\infty} f_n(t) = f(t)$. The limit $\lim_{n\to\infty} f_n(t)$ is called the *Fourier series* of $f(t)$, and the coefficients a_k, b_k are called the *Fourier coefficients.*

The interpretation of (1) is that the partial Fourier series $f_n(t)$ are increasingly better approximations of $f(t)$ with the sums of sine and cosine waves. This provides a decomposition of any such sufficiently nice f into the sum of sinusoids. All frequencies of the sinusoids involved in the sum are multiples of $2\pi/P$, called the *fundamental frequency* of the system. If $f(t)$ describes a sound, then the Fourier series decomposition of $f(t)$ will tell us how it can be expressed as a superposition of pure sinusoids at

[1]We require that f is Riemann integrable. The conclusion of this theorem might fail at certain points $t \in [0, P]$ if f is not nice enough, but other than exceptional cases, the number of such points is usually finite and quite clear (for example, the Fourier series might not converge to the original function at points of discontinuity).

various different frequencies. The Fourier coefficients are then the amplitudes of the sinusoids of different frequencies (pitches) that sum up to our function.

If we define

$$\hat{f}(s) := \int_0^p f(t) \exp(-2\pi i s t)\, dt, \tag{2}$$

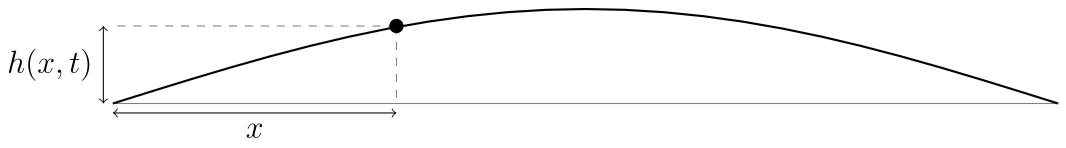
the Fourier series of $f(t)$ may be rewritten as

$$f(t) \overset{*}{=} \frac{1}{P} \sum_{k=-\infty}^{\infty} \hat{f}\left(\frac{k}{P}\right) \exp\left(\frac{2\pi i k t}{P}\right) \tag{3}$$

where the $\overset{*}{=}$ indicates that the two sides might not always be equal everywhere on $[0, P]$. The function $\hat{f}(s)$ is called the *Fourier transform*[2] of $f(t)$, and is sometimes written $\mathcal{F}\{f(t)\}$. Equations (2) and (3) allow us to conveniently decompose $f(t)$ as a Fourier series numerically.

3 Step 1: Finding the PDE

Let us first consider a lower-dimensional case of a system similar to ours. Consider a one-dimensional string of 0 thickness, fixed at both ends. Let x be the position coordinate of the string, with $x \in [0, 1]$. Let $h(x, t)$ be the height of the string at position x at time t.



It is difficult to understand the behaviour of this system, so we turn to finite difference methods. Let us approximate the continuous system by breaking the interval $[0, 1]$ into n equally-spaced intervals, and place $n + 1$ point masses at positions $x = 0, 1/n, \dots, 1$ respectively. We imagine the forces between two particles as being due to springs, and hence governed by Hooke's law (the size of the force is proportional to the distance between the masses).

Let us make a switch of notation and write $h_k(t) = h(k/n, t)$ for the height of the kth mass.

Let us focus on discussing the forces acting on a particular mass, say the kth.

[2]Note that not all functions have a well-defined Fourier transform. The integral in (2) must converge, which forces f to be $O(\exp(2\pi i s t))$ as $t \to 0, P$. In addition, f must also be integrable, along with certain other conditions to guarantee convergence.

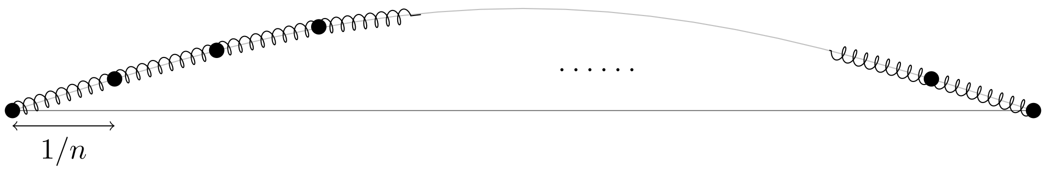


Figure 2: A finite difference approximation with $(n+1)$ point masses.

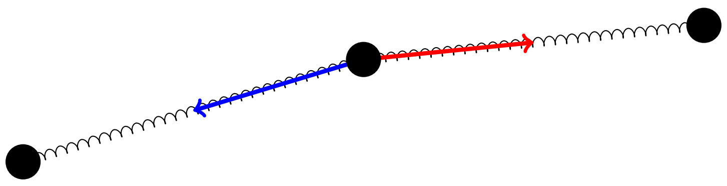


Figure 3: The two spring forces acting on a point mass.

By Hooke's Law, the force due to the $(k+1)$th mass acting on the kth mass is

$$\begin{aligned} F_{k+1,k} &= \underbrace{\left(\frac{1}{\sqrt{(1/n)^2 + (h_k - h_{k+1})^2}}\right)\left(-\frac{1}{n}, -(h_{k+1} - h_k)\right)}_{\text{direction}} \cdot \underbrace{\left(\kappa\sqrt{(1/n)^2 + (h_k - h_{k+1})^2}\right)}_{\text{magnitude}} \\ &= -\kappa\left(\frac{1}{n}, h_{k+1} - h_k\right). \end{aligned}$$

Here, κ is the spring constant, which is equal in both springs. Therefore, the sum of the two forces acting on the kth mass is

$$\begin{aligned} {\color{blue}-F_{k,k-1}} + {\color{red}F_{k+1,k}} &= \kappa\left(\frac{1}{n}, h_k - h_{k-1}\right) - \kappa\left(\frac{1}{n}, h_{k+1} - h_k\right) \\ &= -\kappa\big(0, (h_{k+1} - h_k) - (h_k - h_{k-1})\big). \end{aligned}$$

By Newton's law, this quantity is equal to the mass of the particle times its acceleration. We assume the particle has unit mass, so that we get

$$\frac{d^2}{dt^2} h_k(t) = -\kappa\big((h_{k+1} - h_k) - (h_k - h_{k-1})\big).$$

Now, we simply let $n \to \infty$ to obtain

$$\frac{\partial^2}{\partial t^2} h(x,t) = -\kappa \frac{\partial^2}{\partial x^2} h(x,t). \tag{4}$$

This logic exactly carries over to the higher-dimensional case where we have a two-dimensional disk instead of a one-dimensional string. If we account for possible external forces, we obtain the PDE we need to solve.

$$\boxed{\frac{\partial^2}{\partial t^2}h(\mathbf{x},t) = -\kappa\nabla^2 h(\mathbf{x},t) + F_{\text{ext}}(\mathbf{x},t) \qquad (\star)}$$

Here, $\mathbf{x} \in \mathbb{R}^2$ is the position vector.

It is worth noting that in the case where we have rotational symmetry in the system, we do not need two position variables $(x,y) \in \mathbb{R}^2$ to completely describe our PDE, since all the information can be encoded in a single variable $r := \sqrt{x^2+y^2}$. This allows us to reduce the problem to the lower-dimensional case, since we have that for any F,

$$\frac{\partial F}{\partial x} = \frac{\partial F}{\partial r}\frac{\partial r}{\partial x} = \frac{\partial F}{\partial r}\cos\theta \quad \text{and} \quad \frac{\partial F}{\partial y} = \frac{\partial F}{\partial r}\frac{\partial r}{\partial y} = \frac{\partial F}{\partial r}\sin\theta,$$

where $\theta = \tan^{-1}(y/x)$. Hence

$$\left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right)F = \frac{\partial}{\partial r}\left(\frac{\partial F}{\partial r}\cos\theta\right)\cos\theta + \frac{\partial}{\partial r}\left(\frac{\partial F}{\partial r}\sin\theta\right)\sin\theta = \frac{\partial^2}{\partial r^2}F.$$

Therefore, when there are no external forces, the right-hand side of $(\star)$ reduces to that of (4) in the special case $F = h$.

The only problem with this differential equation is that we do not know what F_{ext} is. We assume from now on that:

$$F_{\text{ext}}(\mathbf{x},t) = \begin{cases} \alpha v, & |\mathbf{x}| < \epsilon \\ 0, & |\mathbf{x}| \geq \epsilon \end{cases}$$

where α, β are positive constants depending on the physical qualities of the system, ϵ is a small positive constant (the radius of the stream of water), and t_0 is a positive real number that indicates the time it takes before the water fills up to form a thick enough layer for the stream of water to hit before impacting the base of the cylinder.

4 Step 2: Solving the PDE

In order to numerically solve the partial differential equation $(\star)$, we will use the program *Wolfram Mathematica*. The full working code can be found in Appendix A.

Code snippet 4.1: Numerically solving the PDE

```
In[1]:=  soln = NDSolveValue[{D[h[x, t], t, t] == k D[h[x, t], x,
x], h[x, 0] == Cos[x Pi/2], h[-1, t] == h[1, t] == 0}, h, {x, -1,
1}, {t, 0, 10}]
```

First, information about the partial differential equation $(\star)$ is encoded into Mathematica's `NDSolveValue` function, which numerically solves the equation given a set of initial conditions. The output is in the form of an interpolating function, consisting of a finite set of approximated data points and the values of the other points approximated (interpolated) based on the data set, given by `soln`. This is the approximate numerical solution to the given PDE.

Code snippet 4.2: Animating the solution

```
In[2]:=  Table[Plot[soln[x, t], {x, -1, 1}, PlotRange -> {-1, 1}],
{t , 0, 10, 0.1}]
```

Then, Mathematica plots the graph of the numerical solution `soln[x,t]` against the position coordinate x for 101 values of the time coordinates `t = 0, 0.1, 0.2, ..., 9.9, 10.0`. This produces a list of 101 plots corresponding to the 101 time coordinates, which can then be compiled into a video by treating them as distinct frames, for the purpose of visualisation.[3]

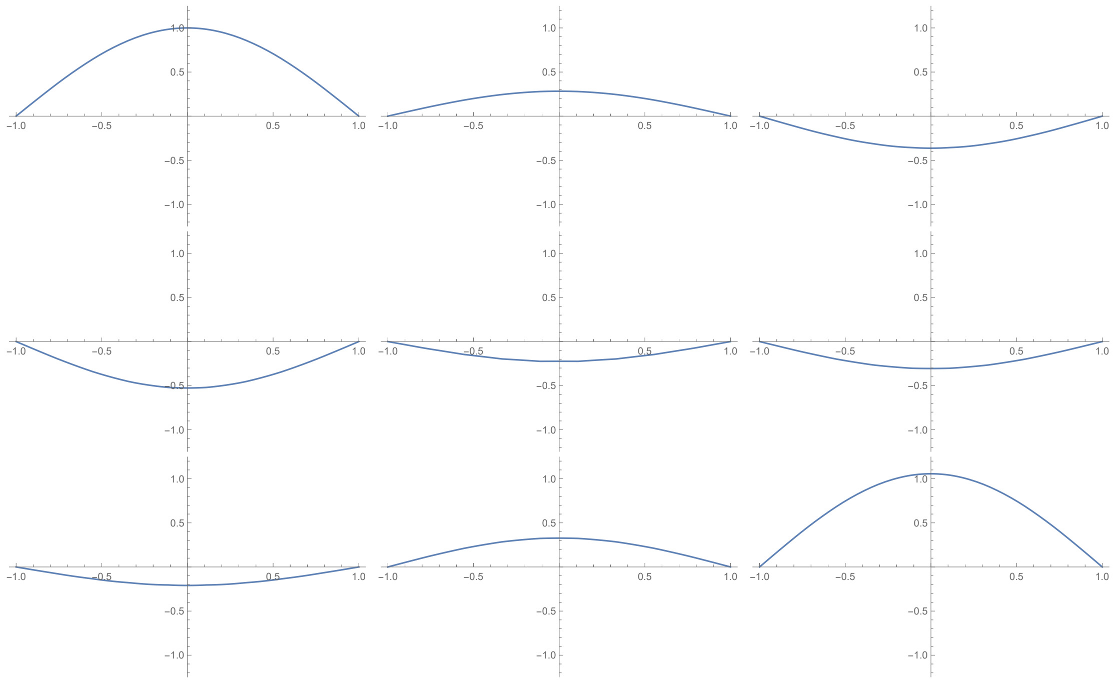


Figure 4: The plots of the approximate numerical solution to the system when $\kappa = 100$, at time coordinates $\texttt{t} = 0, 1, \ldots, 8$. (Left to right, top to bottom.)

[3]For the video in this case, see https://figshare.com/s/1604d3a41f2d19b0d1d9.

5 Step 3: Finding the Fourier coefficients

We now have numerically found a function $h : [-1, 1] \times \mathbb{R}_{\geq 0} \to \mathbb{R}$ which satisfies the PDE. The problem is to now decompose this into a continuous sum of sinusoids corresponding to the pitch at each frequency. Our problem is simplified dramatically by observing that the period of the oscillation (the minimum $P > 0$ such that $h(x, t) = h(x, t+P)$ for all x, t) is independent of x, so we might as well assume that $x = 0$, which reduces the problem to the study of a single variable function $h(0, t)$. For simplicity of notation we will simply refer to this as $h(t)$.

The Fourier transform of h is given by:

$$(\mathcal{F}h)(s) = \int_0^\infty h(t)\exp(-2\pi i s t)dt$$

which corresponds to the amplitude of the sound at frequency s. To find the average frequency, we want to find

$$\lim_{M\to\infty}\left(\frac{1}{M}\int_0^M s\int_0^\infty h(t)\exp(-2\pi i s t)\,dt\,ds\right).$$

This can be achieved by the Mathematica code below.

Code snippet 5.1: Summing the weighted Fourier coefficients

```
In[3]:= Limit[(1/M)NIntegrate[s NIntegrate[h[0, t]Exp[-2Pi i s t],
{t, 0, Infinity}], {s, 0, M}], M->Infinity]
```

This returns a finite numerical value as the answer, resolving our problem.[4] Now, we are left with qualitatively seeing how this quantity depends on our spring constant κ which depends directly on d, and the force F_{ext}, which depends on R and v.

6 Results

The reason that the frequency S depends on the parameters d, R, v is because it depends on κ, which depends on d, and on the external force F_{ext}, which depends on R and v. In the definition of F_{ext} above, we may assume that $\epsilon = 1/R$ since our system is scale-invariant. Choosing units, we may also assume that the magnitude of the force F_{ext} is exactly v (that is, $\alpha = 1$ in the original definition) and that $d = \kappa$.

Therefore, to investigate how S depends on d, R and v, we just need to plot S against each of the variables while cycling through different constant values of the others.

[4]Due to constraints in computing power and time, this code will not run quickly enough to give a precise answer. Instead we compute integer points along the curve of the smooth function and take their weighted average to obtain a good approximation. For the detailed code, see the Appendix.

After cycling through many different constant values of the parameters that are held constant, the shape of the graphs of S against each parameter $\epsilon = 1/R, \kappa = d$ and v are found to be the same. Shown below are representative plots of S against each parameter for specific chosen constant values of the others.

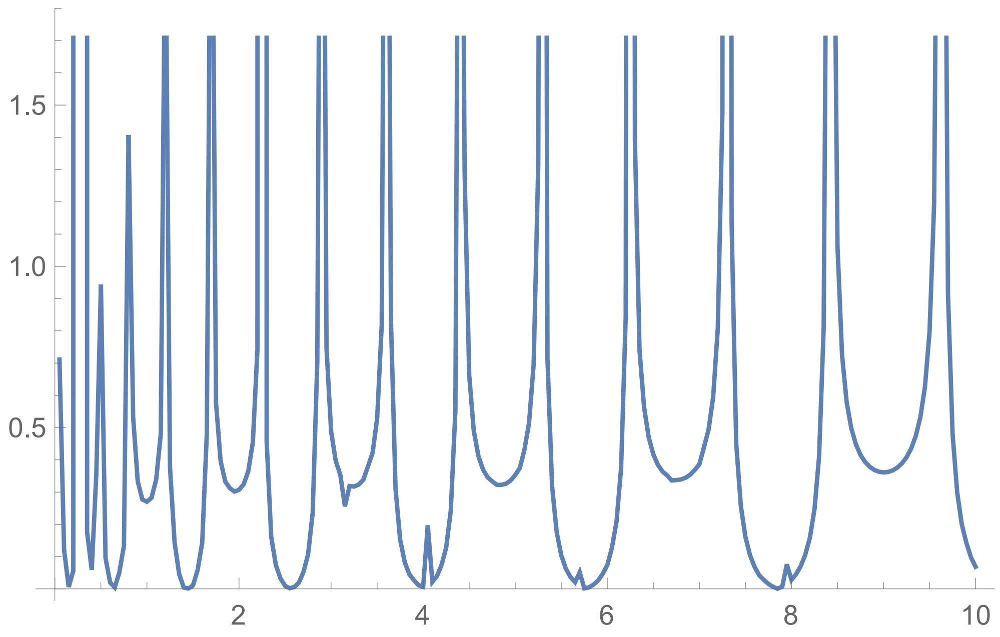


Figure 5: The graph of S against κ, for fixed $v = 1$, $\epsilon = 0.2$. Precision is $\Delta\kappa = 0.05$.

Figure 5 demonstrates a rather counter-intuitive trend in how S depends on κ, which physically corresponds to the thickness of the material. At certain (irregular) intervals, the frequency appears to blow up, giving the graph infinitely many vertical asymptotes. Between these asymptotes, the frequency reaches a minimum point at the middle, with the magnitude of the minimum point alternating between 0 and approximately 0.3.

A plausible physical explanation for this phenomenon is that there is a certain resonant frequency f for the simulated material. At multiples of this frequency, the net forces on each "particle" (in the approximate finite difference system) interfere constructively and increase the magnitude of the oscillations. Thus, the Fourier transform of such a position function will produce magnitudes that also go to infinity, so that the continuous sum over all possible frequencies goes to infinity as well. The asymptotes therefore occur at values of κ that correspond to multiples of f.

From Figure 6, there is a clear decreasing trend in S as v increases. The trend is approximately linear, but appears to be slightly convex. At around $v = 25$ and $v = 65$, there is a sharp nonlinear change in the values of S corresponding to a small change in v. This phenomenon should not be interpreted as having any physical significance; the most likely cause for this observation is that in the interest of speeding up the computation, we used a discrete approximation to an integral with a sum (see

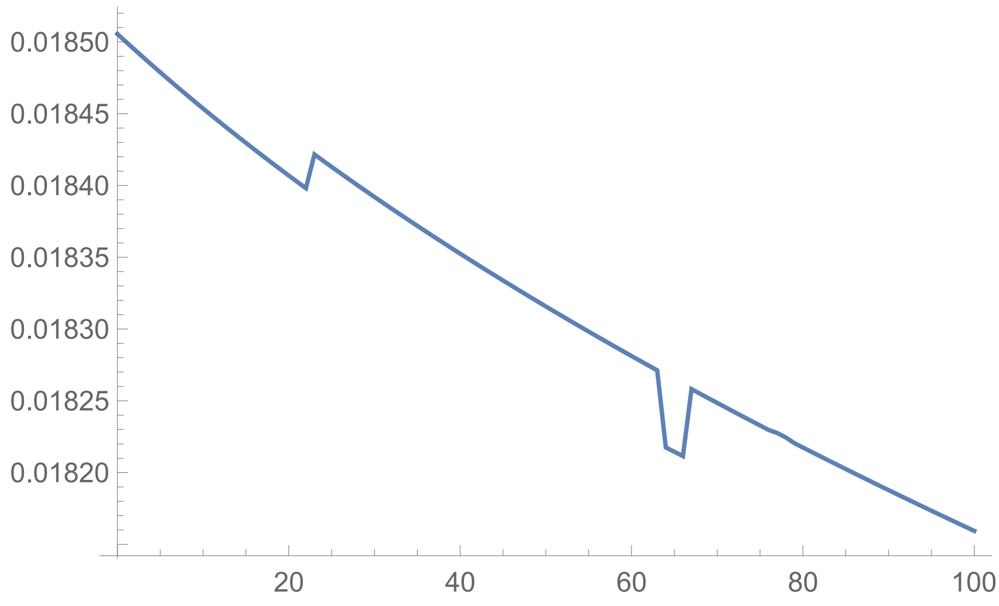


Figure 6: The graph of S against v, for fixed $\kappa = 100$, $\epsilon = 0.2$. Precision is $\Delta v = 1$.

Appendix A). The discrete nature of the summation to approximate the theoretically continuous integral might have led to these discrepancies occurring.

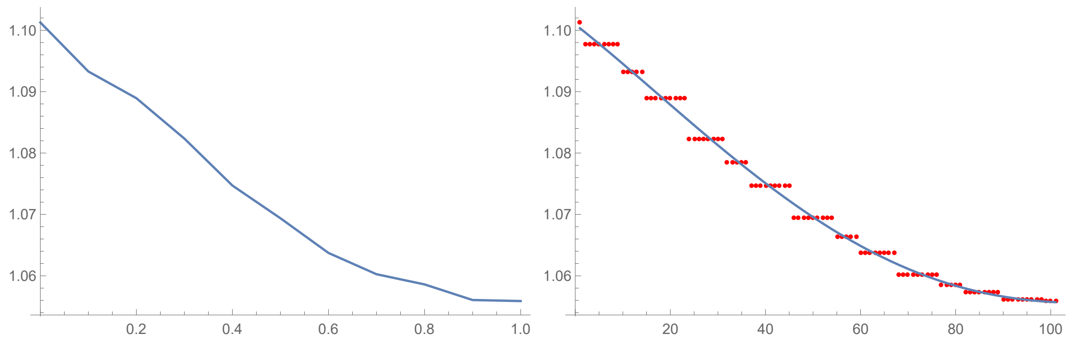


Figure 7: (Left) The graph of S against ϵ, for fixed $\kappa = 125$, $v = 10$. Precision is $\Delta\epsilon = 0.1$. (Right) The fourth order Taylor fit to the data points.

Figure 7 shows the graph of S against ϵ. Again, there is an obvious nonlinear decreasing trend as ϵ increases, corresponding to the stream of water becoming wider in diameter. This makes sense because when ϵ is lower, a smaller part of the base is affected by the external force F_{ext}, which leads to a greater difference in h at the border $|\mathbf{x}| = \epsilon$, and hence a greater resultant force and greater amplitude and frequency of oscillations.

According to Mathematica, the fourth order Taylor fit to the given data points is

$$1.1009 - 6.1229 \times 10^{-4}x - 3.5749 \times 10^{-6}x^2 + 7.9014 \times 10^{-8}x^3 - 2.7225 \times 10^{-10}x^4.$$

7 Conclusion

In conclusion, this project has created an analytical model of the desired system using a partial differential equation, solved the equation and obtained the numerical solution to a high degree of precision. The trends predicted by our model agree with our intuition of what should happen as we change the parameters involved in the question.

With more computing power and more time, further research might look into computing our models with higher degrees of precision for a graph of higher resolution. Alternative ways to model our system (for example using a Lagrangian formulation) might also be of interest. Higher dimensional analogues of our problem might also be an interesting extension of our work.

Appendix A Mathematica Code

The following is the full *Mathematica* code that was used to produce the results provided in the previous sections. The authors used Mathematica version 11.0.1.0.

A.1: Producing the animated solution

```
In[1]:= e := 0.1; k := 100; v := 1;
  soln = NDSolveValue[{D[h[x, t], t, t] == k D[h[x, t], x, x] +
  Piecewise[{{v, Abs[x] < e}}], h[x, 0] == Cos[x Pi/2], h[-1, t]
  == h[1, t] == 0}, h, {x, -1, 1}, {t, 0, 10}];
  Export["video.avi", Table[Plot[soln[x, t], {x, -1, 1}, PlotRange
-> {-1, 1}], {t, 0, 1, 0.01}]]
```

A.2: Producing the S–κ plot

```
In[1]:= ResLst = {};
Do[{ e := 0.2; v := 1;
  soln = NDSolveValue[{D[h[x, t], t, t] == k D[h[x, t], x, x] +
  Piecewise[{{v, Abs[x] < e}}], h[x, 0] == Cos[x Pi/2], h[-1, t]
  == h[1, t] == 0}, h, {x, -1, 1}, {t, 0, 10}];
  Fsoln[s_, M_] := NIntegrate[soln[0, t] Exp[-2 Pi i s t],
  {t, 0, M}, WorkingPrecision -> 10];
  Ans[M_] := Sum[s Abs[Fsoln[s, 10]], s, 0, M, 1]/M;
  AppendTo[ResLst, Ans[10]], Print[k]}, {k, 0, 10, 0.1}]
ListPlot[Transpose[{Range[0, 10, 0.1], ResLst}], Joined -> True]
```

A.3: Producing the S–v plot

```
In[1]:= ResLst = {};
Do[{ e := 0.2; k := 100;
  soln = NDSolveValue[{D[h[x, t], t, t] == k D[h[x, t], x, x] +
  Piecewise[{{v, Abs[x] < e}}], h[x, 0] == Cos[x Pi/2], h[-1, t]
  == h[1, t] == 0}, h, {x, -1, 1}, {t, 0, 10}];
  Fsoln[s_, M_] := NIntegrate[soln[0, t] Exp[-2 Pi i s t],
  {t, 0, M}, WorkingPrecision -> 10];
  Ans[M_] := Sum[s Abs[Fsoln[s, 10]], s, 0, M, 1]/M;
  AppendTo[ResLst, Ans[10]], Print[v]}, {v, 0, 100, 1}]
ListPlot[Transpose[{Range[0, 100, 1], ResLst}], Joined -> True]
```

A.4: Producing the S–ϵ plot

```
In[1]:= ResLst = {};
Do[{ v := 10; k := 125;
  soln = NDSolveValue[{D[h[x, t], t, t] == k D[h[x, t], x, x] +
  Piecewise[{{v, Abs[x] < e}}], h[x, 0] == Cos[x Pi/2], h[-1, t]
  == h[1, t] == 0}, h, {x, -1, 1}, {t, 0, 10}];
  Fsoln[s_, M_] := NIntegrate[soln[0, t] Exp[-2 Pi i s t],
  {t, 0, M}, WorkingPrecision -> 10];
  Ans[M_] := Sum[s Abs[Fsoln[s, 10]], s, 0, M, 1]/M;
  AppendTo[ResLst, Ans[10]], Print[e]}, {e, 0, 1, 0.1}]
ListPlot[Transpose[{Range[0, 1, 0.1], ResLst}], Joined -> True]
```

References

[1] Swigon, D. (2011). *Math 1360: Modelling in Applied Math I, Lecture 8 Notes, Molecular Modelling.* Retrieved from http://www.math.pitt.edu/ swigon/Lectures/Lecture8.pdf .

[2] Smith, G. D. (1985),Numerical Solution of Partial Differential Equations: Finite Difference Methods, 3rd ed., Oxford University Press

[3] Iserles, A. (2008).A first course in the numerical analysis of differential equations. Cambridge University Press. p.23.ISBN9780521734905.

[4] Fourier, J. (1807). Mmoire sur la propagation de la chaleur dans les corps solides. In J. Darboux (Ed.),Oeuvres de Fourier: Publies par les soins de Gaston Darboux(Cambridge Library Collection - Mathematics, pp. 213-222). Cambridge: Cambridge University Press. doi:10.1017/CBO9781139568159.009

[5] Lagrange, J. (1770). Rflexions sur la rsolution algbrique des quations. Nouveaux mmoires de l'Acadmie royale des sciences et belles-lettres de Berlin, 1770-1771 (New memoirs of the Royal Academy of Sciences and belles-lettres of Berlin).