CS7GV2: Mathematics of Light and Sound, M.Sc. in Computer Science.

Lecture #3: Simulation

Fergal Shevlin, Ph.D.

School of Computer Science and Statistics,
Trinity College Dublin

October 11, 2024

Wave Motion

We've seen that wave motion is described by the second order PDE known as the wave equation,

$$\frac{\partial^2 A(x,t)}{\partial t^2} = c^2 \frac{\partial^2 A(x,t)}{\partial x^2}.$$

We've seen a *closed-form* solution for wave propagation,

$$A(x,t) = R\cos(kx - \omega t) + (1-R)\cos(kx + \omega t).$$

This is perfect in certain conditions, e.g. light in a homogeneous medium, a wave on an infinitely long string, or a sound in a huge volume of air.

But it doesn't tell us, for example, how a string plucked in a particular way is going to move: `https://tinyurl.com/y4ncymx7`.

Or the effects of wave superposition, reflection, refraction, diffraction, etc.

Wave Motion

We've seen that wave motion is described by the second order PDE known as the wave equation,

$$\frac{\partial^2 A(x,t)}{\partial t^2} = c^2 \frac{\partial^2 A(x,t)}{\partial x^2}.$$

We've seen a *closed-form* solution for wave propagation,

$$A(x,t) = R\cos(kx - \omega t) + (1 - R)\cos(kx + \omega t).$$

This is perfect in certain conditions, e.g. light in a homogeneous medium, a wave on an infinitely long string, or a sound in a huge volume of air.

But it doesn't tell us, for example, how a string plucked in a particular way is going to move: `https://tinyurl.com/y4ncymx7`.

Or the effects of wave superposition, reflection, refraction, diffraction, etc.

Wave Motion

We've seen that wave motion is described by the second order PDE known as the wave equation,

$$\frac{\partial^2 A(x,t)}{\partial t^2} = c^2 \frac{\partial^2 A(x,t)}{\partial x^2}.$$

We've seen a *closed-form* solution for wave propagation,

$$A(x,t) = R\cos(kx - \omega t) + (1 - R)\cos(kx + \omega t).$$

This is perfect in certain conditions, e.g. light in a homogeneous medium, a wave on an infinitely long string, or a sound in a huge volume of air.

But it doesn't tell us, for example, how a string plucked in a particular way is going to move: `https://tinyurl.com/y4ncymx7`.

Or the effects of wave superposition, reflection, refraction, diffraction, etc.

Wave Motion

We've seen that wave motion is described by the second order PDE known as the wave equation,

$$\frac{\partial^2 A(x,t)}{\partial t^2} = c^2 \frac{\partial^2 A(x,t)}{\partial x^2}.$$

We've seen a *closed-form* solution for wave propagation,

$$A(x,t) = R\cos(kx - \omega t) + (1-R)\cos(kx + \omega t).$$

This is perfect in certain conditions, e.g. light in a homogeneous medium, a wave on an infinitely long string, or a sound in a huge volume of air.

But it doesn't tell us, for example, how a string plucked in a particular way is going to move: `https://tinyurl.com/y4ncymx7`.

Or the effects of wave superposition, reflection, refraction, diffraction, etc.

Wave Motion

We've seen that wave motion is described by the second order PDE known as the wave equation,

$$\frac{\partial^2 A(x,t)}{\partial t^2} = c^2 \frac{\partial^2 A(x,t)}{\partial x^2}.$$

We've seen a *closed-form* solution for wave propagation,

$$A(x,t) = R\cos(kx - \omega t) + (1-R)\cos(kx + \omega t).$$

This is perfect in certain conditions, e.g. light in a homogeneous medium, a wave on an infinitely long string, or a sound in a huge volume of air.

But it doesn't tell us, for example, how a string plucked in a particular way is going to move: `https://tinyurl.com/y4ncymx7`.

Or the effects of wave superposition, reflection, refraction, diffraction, etc.

Wave Motion

We've seen that wave motion is described by the second order PDE known as the wave equation,

$$\frac{\partial^2 A(x,t)}{\partial t^2} = c^2 \frac{\partial^2 A(x,t)}{\partial x^2}.$$

We've seen a *closed-form* solution for wave propagation,

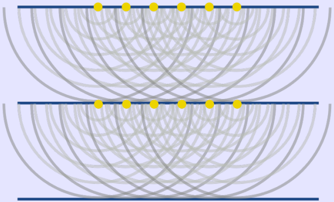
$$A(x,t) = R\cos(kx - \omega t) + (1 - R)\cos(kx + \omega t).$$

This is perfect in certain conditions, e.g. light in a homogeneous medium, a wave on an infinitely long string, or a sound in a huge volume of air.

But it doesn't tell us, for example, how a string plucked in a particular way is going to move: `https://tinyurl.com/y4ncymx7`.

Or the effects of wave superposition, reflection, refraction, diffraction, etc.

Wavefront propagation



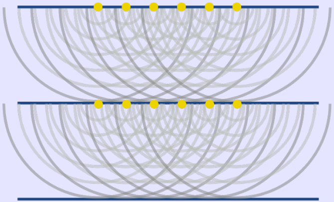
Huygens-Fresnel Construction.

Refraction.

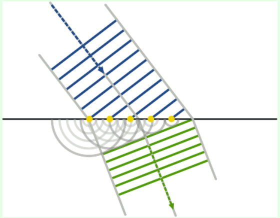
Reflection.

Diffraction.

Wavefront propagation



Huygens-Fresnel Construction.

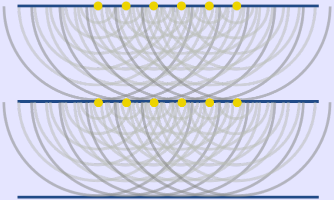


Refraction.

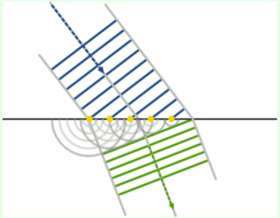
Reflection.

Diffraction.

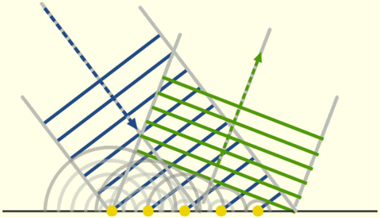
Wavefront propagation



Huygens-Fresnel Construction.



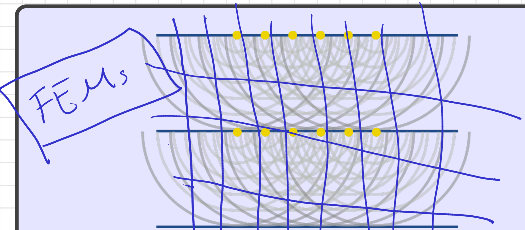
Refraction.



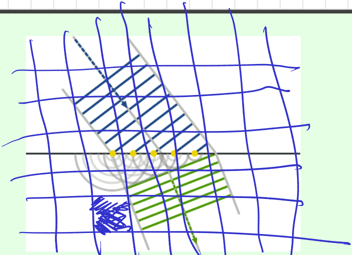
Reflection.

Diffraction.

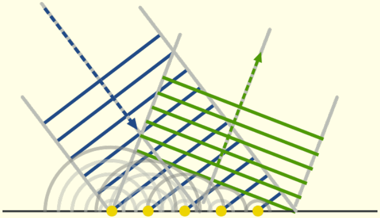
Wavefront propagation



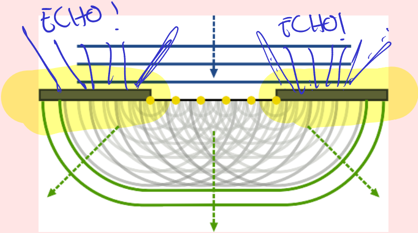
Huygens-Fresnel Construction.



Refraction.



Reflection.



Diffraction.

Numerical methods

The roots (zero-crossings) of a quadratic polynomial $f(x) = a\,x^2 + b\,x + c$, can be found as,

$$x_1, x_2 = \frac{-b \pm \sqrt{b^2 - 4\,ac}}{2\,a}.$$

What are the roots of $g(x) = a\,x^4 + b\,x^2 + c$?

In practice, most mathematical problems don't have *analytical* or even *closed-form* solutions.

You might remember the "method of bisection:" guess where a root might be; keep halving the length of an interval around it such that $f(x)$ has different signs at the start and the end.

Such an approach is described as a *numerical method* because it uses numerical approximations, typically over multiple iterations, to find a solution.

Floating-point numbers on a computer have finite precision so error is inevitable.

If x is a true solution and $\tilde{x}$ is a calculated one, the error $\epsilon_x = x - \tilde{x}$.

Relative error $|\epsilon_x|/|x|$ is more informative.

For $x - y = \tilde{x} + \epsilon_x - \tilde{y} - \epsilon_y$, the relative error $|\epsilon_x - \epsilon_y|/|x - y|$ is large for a small difference.

Error is typically cumulative so the results become less correct at each iteration.

There are nice interactive simulations using numerical methods available, for example:

`https://tinyurl.com/2xrsrz` and `https://tinyurl.com/mtwczmj`.

Eventually these should all fail!

Numerical methods

* "Hidden figures" *

The roots (zero-crossings) of a quadratic polynomial $f(x) = a\,bx^2 + b\,x + c$, can be found as,

$$x_1, x_2 = \frac{-b \pm \sqrt{b^2 - 4\,ac}}{2\,a}.$$

What are the roots of $g(x) = a\,x^4 + b\,x^2 + c$?

In practice, most mathematical problems don't have *analytical* or even *closed-form* solutions.

You might remember the "method of bisection:" guess where a root might be; keep halving the length of an interval around it such that $f(x)$ has different signs at the start and the end.

Such an approach is described as a *numerical method* because it uses numerical approximations, typically over multiple iterations, to find a solution.

Floating-point numbers on a computer have finite precision so error is inevitable.

If x is a true solution and $\tilde{x}$ is a calculated one, the error $\epsilon_x = x - \tilde{x}$.

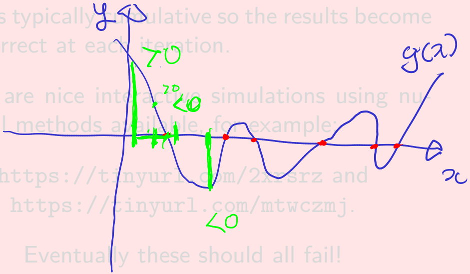
Relative error $|\epsilon_x|/|x|$ is more informative.

For $x - y = \tilde{x} + \epsilon_x - \tilde{y} - \epsilon_y$, the relative error $|\epsilon_x - \epsilon_y|/|x - y|$ is large for a small difference.

Error is typically cumulative so the results become less correct at each iteration.

There are nice interactive simulations using numerical methods available, for example:

https://tinyurl.com/2xrsrz and https://tinyurl.com/mtwczmj.



Eventually these should all fail!

Numerical methods

The roots (zero-crossings) of a quadratic polynomial $f(x) = a\,bx^2 + b\,x + c$, can be found as,

$$x_1, x_2 = \frac{-b \pm \sqrt{b^2 - 4\,ac}}{2\,a}.$$

What are the roots of $g(x) = a\,x^4 + b\,x^2 + c$?

In practice, most mathematical problems don't have *analytical* or even *closed-form* solutions.

You might remember the "method of bisection:" guess where a root might be; keep halving the length of an interval around it such that $f(x)$ has different signs at the start and the end.

Such an approach is described as a *numerical method* because it uses numerical approximations, typically over multiple iterations, to find a solution.

Floating-point numbers on a computer have finite precision so error is inevitable.

If $\boxed{x}$ is a true solution and $\boxed{\tilde{x}}$ is a calculated one, the error $\epsilon_x = x - \tilde{x}$.

2/3

2/150,000

Relative error $|\epsilon_x|/|x|$ is more informative.

For $x - y = \boxed{\tilde{x} + \epsilon_x}\ \boxed{- \tilde{y} - \epsilon_y}$, the relative error $|\epsilon_x - \epsilon_y|/|x - y|$ is large for a small difference.

Error is typically cumulative so the results become less correct at each iteration.

There are nice interactive simulations using numerical methods available, for example:

`https://tinyurl.com/2xrsrz` and
`https://tinyurl.com/mtwczmj`.

Eventually these should all fail!

Numerical methods

The roots (zero-crossings) of a quadratic polynomial $f(x) = a\,bx^2 + b\,x + c$, can be found as,

$$x_1, x_2 = \frac{-b \pm \sqrt{b^2 - 4\,ac}}{2\,a}.$$

What are the roots of $g(x) = a\,x^4 + b\,x^2 + c$?

In practice, most mathematical problems don't have *analytical* or even *closed-form* solutions.

You might remember the "method of bisection:" guess where a root might be; keep halving the length of an interval around it such that $f(x)$ has different signs at the start and the end.

Such an approach is described as a *numerical method* because it uses numerical approximations, typically over multiple iterations, to find a solution.

Floating-point numbers on a computer have finite precision so error is inevitable.

If x is a true solution and $\tilde{x}$ is a calculated one, the error $\epsilon_x = x - \tilde{x}$.

Relative error $|\epsilon_x|/|x|$ is more informative.

For $x - y = \tilde{x} + \epsilon_x - \tilde{y} - \epsilon_y$, the relative error $|\epsilon_x - \epsilon_y|/|x - y|$ is large for a small difference.

Error is typically cumulative so the results become less correct at each iteration.

There are nice interactive simulations using numerical methods available, for example:

`https://tinyurl.com/2xrsrz` and
`https://tinyurl.com/mtwczmj`.

Eventually these should all fail!

Finite Element Methods

→ Modelling combustion in car ICE
→ Modelling a beam in structural eng.
→ Weather forecasting.
→ Heatsinks

Consider space as a finite number of discrete elements, e.g. little cubes. And consider time as a finite number of intervals.

Solve the appropriate equation (typically differential) to find the characteristics of interest at each element of space at successive steps in time.

The characteristics for one discrete volume at one time interval are used in the calculation of the characteristics of its neighbors at the next time interval.

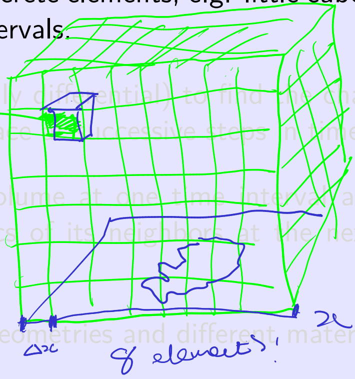
Advantages: can deal with complex geometries and different materials.

Disadvantages: can be very computationally intensive which limits the spatial accuracy or the temporal duration, cf. weather forecasting.

Finite Element Methods

Consider space as a "finite" number of discrete elements, e.g. little cubes. And consider time as a finite number of intervals.

element



space Δx

time Δt

Solve the appropriate equation (typically differential) to find the characteristics of interest at each element of space at successive steps in time.

The characteristics for one discrete volume at one time interval are used in the calculation of the characteristics of its neighbors at the next time interval.

Advantages: can deal with complex geometries and different materials.

Disadvantages: can be very computationally intensive which limits the spatial accuracy or the temporal duration, cf. weather forecasting.

Finite Element Methods

Consider space as a finite number of discrete elements, e.g. little cubes. And consider time as a finite number of intervals. wave eq.

Solve the appropriate equation (typically differential) to find the characteristics of interest at each element of space at successive steps in time.

The characteristics for one discrete volume at one time interval are used in the calculation of the characteristics of its neighbors at the next time interval.

Advantages: can deal with complex geometries and different materials.

Disadvantages: can be very computationally intensive which limits the spatial accuracy or the temporal duration, cf. weather forecasting.

Finite Element Methods

Consider space as a finite number of discrete elements, e.g. little cubes. And consider time as a finite number of intervals.

Solve the appropriate equation (typically differential) to find the characteristics of interest at each element of space at successive steps in time.

The characteristics for one discrete volume at one time interval are used in the calculation of the characteristics of its neighbors at the next time interval.

Advantages: can deal with complex geometries and different materials.

Disadvantages: can be very computationally intensive which limits the spatial accuracy or the temporal duration, cf. weather forecasting.

Finite Element Methods

Consider space as a finite number of discrete elements, e.g. little cubes. And consider time as a finite number of intervals.

Solve the appropriate equation (typically differential) to find the characteristics of interest at each element of space at successive steps in time.

The characteristics for one discrete volume at one time interval are used in the calculation of the characteristics of its neighbors at the next time interval.

Advantages: can deal with complex geometries and different materials.

Disadvantages: can be very computationally intensive which limits the spatial accuracy or the temporal duration, cf. weather forecasting.

Finite Element Methods

Consider space as a finite number of discrete elements, e.g. little cubes. And consider time as a finite number of intervals.

Solve the appropriate equation (typically differential) to find the characteristics of interest at each element of space at successive steps in time.

The characteristics for one discrete volume at one time interval are used in the calculation of the characteristics of its neighbors at the next time interval.

Advantages: can deal with complex geometries and different materials.

Disadvantages: can be very computationally intensive which limits the spatial accuracy or the temporal duration, cf. weather forecasting.

"super computers"

Initial and Boundary Conditions

To simulate a specific solution for $A(x, t)$ described by the wave equation,

$$\frac{\partial^2 A(x,t)}{\partial t^2} = c^2 \frac{\partial^2 A(x,t)}{\partial x^2} \quad x \in [0, L], \ t \in [0, T],$$

for a string of length L over a time period T, we need:

two *initial conditions* at time $t = 0$,

$$A(x, 0) = I(x), \quad x \in [0, L]$$

$$\frac{\partial}{\partial t} A(x, 0) = 0, \quad x \in [0, L]$$

where $I(x)$ specifies the initial shape of the string,

and two *boundary conditions* at distances $x = 0$ and $x = L$,

$$A(0, t) = 0, \quad t \in [0, T]$$

$$A(L, t) = 0, \quad t \in [0, T]$$

Initial and Boundary Conditions

To simulate a specific solution for $A(x,t)$ described by the wave equation,

$$\frac{\partial^2 A(x,t)}{\partial t^2} = c^2 \frac{\partial^2 A(x,t)}{\partial x^2} \quad x \in [0, L],\ t \in [0, T],$$

for a string of length L over a time period T, we need:

two *initial conditions* at time $t = 0$,

$$A(x,0) = I(x), \quad x \in [0, L]$$

$$\frac{\partial}{\partial t} A(x,0) = 0, \quad x \in [0, L]$$

where $I(x)$ specifies the initial shape of the string,

and two *boundary conditions* at distances $x = 0$ and $x = L$,

$$A(0,t) = 0, \quad t \in [0, T]$$

$$A(L,t) = 0, \quad t \in [0, T]$$

Initial and Boundary Conditions

To simulate a specific solution for $A(x,t)$ described by the wave equation,

$$\frac{\partial^2 A(x,t)}{\partial t^2} = c^2 \frac{\partial^2 A(x,t)}{\partial x^2} \quad x \in [0,L],\ t \in [0,T],$$

for a string of length L over a time period T, we need:

two *initial conditions* at time $t = 0$,

$$A(x,0) = I(x), \quad x \in [0,L]$$

Initial shape of string.

$$\frac{\partial}{\partial t} A(x,0) = 0, \quad x \in [0,L]$$

string is not moving!

where $I(x)$ specifies the initial shape of the string,

and two *boundary conditions* at distances $x = 0$ and $x = L$,

$$A(0,t) = 0, \quad t \in [0,T]$$
$$A(L,t) = 0, \quad t \in [0,T]$$

Initial and Boundary Conditions

To simulate a specific solution for $A(x, t)$ described by the wave equation,

$$\frac{\partial^2 A(x,t)}{\partial t^2} = c^2 \frac{\partial^2 A(x,t)}{\partial x^2} \quad x \in [0, L],\ t \in [0, T],$$

for a string of length L over a time period T, we need:

two *initial conditions* at time $t = 0$,

$$A(x, 0) = I(x), \quad x \in [0, L]$$

$$\frac{\partial}{\partial t} A(x, 0) = 0, \quad x \in [0, L]$$

where $I(x)$ specifies the initial shape of the string,

and two *boundary conditions* at distances $x = 0$ and $x = L$,

$$A(0, t) = 0, \quad t \in [0, T]$$

$$A(L, t) = 0, \quad t \in [0, T]$$

Discretization of domain

Computer operations take a finite amount of time to complete so there can't be infinitely many time steps in the simulation.

The time period $[0, T]$ has to be descretized, e.g. into intervals of equal duration Δt,

$$t_i = i\,\Delta t, \quad i = 0, \ldots N_t \text{ (where } N_t = T/\Delta t.\text{)}$$

Computer memory is finite so there can't be infinitely many distances in the simulation.

The length $[0, L]$ have to be descretized, e.g. into intervals of equal distance Δx,

$$x_j = j\,\Delta x, \quad j = 0, \ldots N_x \text{ (where } N_x = L/\Delta x.\text{)}$$

Discretization of domain

Computer operations take a finite amount of time to complete so there can't be infinitely many time steps in the simulation.

The time period $[0, T]$ has to be descretized, e.g. into intervals of equal duration Δt,

$$t_i = i\,\Delta t, \quad i = 0, \ldots N_t \text{ (where } N_t = T/\Delta t.\text{)}$$

Computer memory is finite so there can't be infinitely many distances in the simulation.

The length $[0, L]$ have to be descretized, e.g. into intervals of equal distance Δx,

$$x_j = j\,\Delta x, \quad j = 0, \ldots N_x \text{ (where } N_x = L/\Delta x.\text{)}$$

Discretization of domain

Computer operations take a finite amount of time to complete so there can't be infinitely many time steps in the simulation.

The time period $[0, T]$ has to be descretized, e.g. into intervals of equal duration Δt,

$$t_i = i\,\Delta t, \quad i = 0, \ldots N_t \text{ (where } N_t = T/\Delta t.\text{)}$$

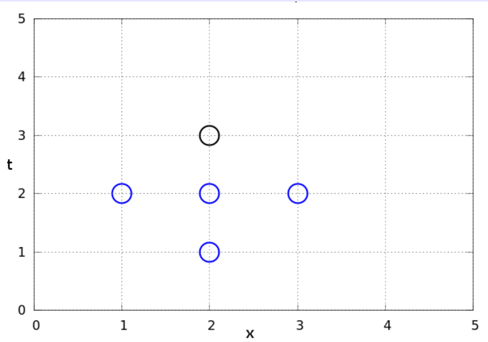
Computer memory is finite so there can't be infinitely many distances in the simulation.

The length $[0, L]$ have to be descretized, e.g. into intervals of equal distance Δx,

$$x_j = j\,\Delta x, \quad j = 0, \ldots N_x \text{ (where } N_x = L/\Delta x.\text{)}$$

Solution mesh

Discrete points in space and time can be visualized as a *mesh*.

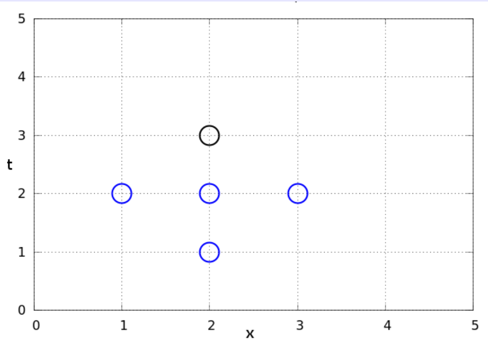


The solution for wave height $A(x_j, t_i)$ at each mesh point is found using already-calculated solutions at neighbouring mesh points ...

...except for certain exterior mesh points whose values have been specified through the initial conditions, i.e. $I(x)$.

Solution mesh

Discrete points in space and time can be visualized as a *mesh*.

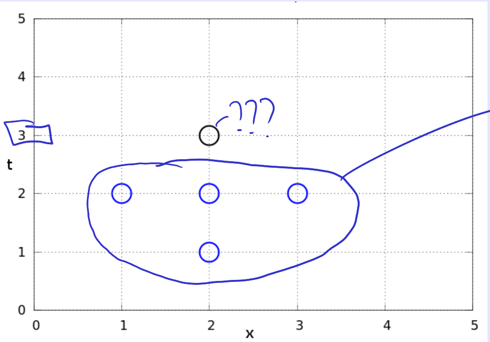


The solution for wave height $A(x_j, t_i)$ at each mesh point is found using already-calculated solutions at neighbouring mesh points ...

... except for certain exterior mesh points whose values have been specified through the initial conditions, i.e. $I(x)$.

Solution mesh

Discrete points in space and time can be visualized as a *mesh*.



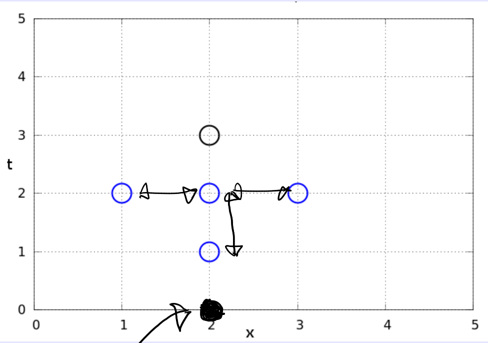
Already calculated before time t=3

The solution for wave height $A(x_j, t_i)$ at each mesh point is found using already-calculated solutions at neighbouring mesh points . . .

. . . except for certain exterior mesh points whose values have been specified through the initial conditions, i.e. $I(x)$.

Solution mesh

Discrete points in space and time can be visualized as a *mesh*. (net)



The solution for wave height $A(x_j, t_i)$ at each mesh point is found using already-calculated solutions at neighbouring mesh points . . .

. . . except for certain exterior mesh points whose values have been specified through the initial conditions, i.e. $I(x)$.

Discretization of equations

Discrete approximations of second derivatives

Symmetric second difference approximates the second derivative so the wave equation becomes,

$$\frac{\partial^2 A(x,t)}{\partial t^2} \approx \frac{\overbrace{A(x_j,t_{i+1}) - A(x_j,t_i)} + \overbrace{A(x_j,t_{i-1}) - A(x_j,t_i)}}{}$$

$$\approx \frac{A(x_j, t_{i+1}) - 2\,A(x_j, t_i) + A(x_j, t_{i-1})}{\Delta t^2} \approx$$

$$\approx c^2 \frac{A(x_{j+1}, t_i) - 2\,A(x_j, t_i) + A(x_{j-1}, t_i)}{\Delta x^2} \approx c^2 \frac{\partial^2 A(x,t)}{\partial x^2}.$$

sum of 2 differences

Alternative notation can be used to make the parameters more obvious,

$$\frac{a_j^{i+1} - 2a_j^i + a_j^{i-1}}{\Delta t^2} \approx c^2 \frac{a_{j+1}^i - 2a_j^i + a_{j-1}^i}{\Delta x^2}, \tag{1}$$

Discretization of equations

Symmetric second difference approximates the second derivative so the wave equation becomes,

$$\frac{A(x_j, t_{i+1}) - 2\,A(x_j, t_i) + A(x_j, t_{i-1})}{\Delta t^2} \approx$$
$$c^2 \frac{A(x_{j+1}, t_i) - 2\,A(x_j, t_i) + A(x_{j-1}, t_i)}{\Delta x^2}.$$

Alternative notation can be used to make the parameters more obvious,

$$\frac{a_j^{i+1} - 2a_j^i + a_j^{i-1}}{\Delta t^2} \approx c^2 \frac{a_{j+1}^i - 2a_j^i + a_{j-1}^i}{\Delta x^2}, \tag{1}$$

Discretization of equations

Symmetric second difference approximates the second derivative so the wave equation becomes,

$$a_j^{i+1} \equiv \frac{A(x_j, t_{i+1}) - 2\,A(x_j, t_i) + A(x_j, t_{i-1})}{\Delta t^2} \approx c^2 \frac{A(x_{j+1}, t_i) - 2\,A(x_j, t_i) + A(x_{j-1}, t_i)}{\Delta x^2}.$$

Alternative notation can be used to make the parameters more obvious,

$$\frac{a_j^{i+1} - 2a_j^i + a_j^{i-1}}{\Delta t^2} \approx c^2 \frac{a_{j+1}^i - 2a_j^i + a_{j-1}^i}{\Delta x^2}, \tag{1}$$

Initial Conditions

Use the centered first difference approximation of the first derivative,

$$\frac{\partial}{\partial t}a(x_j, t_i) \approx \frac{a_j^{i+1} - a_j^{i-1}}{2\Delta t} \tag{2}$$

Note division by $2\Delta t$ because the difference is between values of $a(x, t)$ separated by two time intervals.

Using approximation (2), initial condition $\frac{\partial}{\partial t}a(x_j, 0) = 0$ means,

$$a_j^{i-1} = a_j^{i+1}, \quad j = 0, \ldots, N_x. \quad i = 0.$$

The intial condition of shape is simply,

$$a_j^0 = I(x_j), \quad j = 0, \ldots, N_x.$$

Initial Conditions

Use the centered first difference approximation of the first derivative,

Numerical approximations of derivatives.

$$\frac{\partial}{\partial t} a(x_j, t_i) \approx \frac{a_j^{i+1} - a_j^{i-1}}{2\Delta t} \tag{2}$$

Note division by $2\Delta t$ because the difference is between values of $a(x, t)$ separated by two time intervals.

Using approximation (2), initial condition $\frac{\partial}{\partial t} a(x_j, 0) = 0$ means,

$$a_j^{i-1} = a_j^{i+1}, \quad j = 0, \ldots, N_x. \quad i = 0.$$

The intial condition of shape is simply,

$$a_j^0 = I(x_j), \quad j = 0, \ldots, N_x.$$

Initial Conditions

Use the centered first difference approximation of the first derivative,

$$\frac{\partial}{\partial t} a(x_j, t_i) \approx \frac{a_j^{i+1} - a_j^{i-1}}{2\Delta t} \tag{2}$$

Note division by $2\Delta t$ because the difference is between values of $a(x,t)$ separated by two time intervals.

Using approximation (2), initial condition $\frac{\partial}{\partial t} a(x_j, 0) = 0$ means,

$$a_j^{i-1} = a_j^{i+1}, \quad j = 0, \ldots, N_x. \quad i = 0.$$

The intial condition of shape is simply,

$$a_j^0 = I(x_j), \quad j = 0, \ldots, N_x.$$

Initial Conditions

Use the centered first difference approximation of the first derivative,

$$\frac{\partial}{\partial t} a(x_j, t_i) \approx \frac{a_j^{i+1} - a_j^{i-1}}{2\Delta t} \tag{2}$$

Note division by $2\Delta t$ because the difference is between values of $a(x, t)$ separated by two time intervals.

Using approximation (2), initial condition $\frac{\partial}{\partial t} a(x_j, 0) = 0$ means,

$$a_j^{i-1} = a_j^{i+1}, \quad j = 0, \ldots, N_x. \quad i = 0.$$

The intial condition of shape is simply,

$$a_j^0 = I(x_j), \quad j = 0, \ldots, N_x.$$

Iterative Simulation Algorithm

The required formulae are,

$$C = c\frac{\Delta t}{\Delta x}.$$

$$a_j^{i+1} = -a_j^{i-1} + 2a_j^i + C^2\left(a_{j+1}^i - 2a_j^i + a_{j-1}^i\right)$$

$$a_j^1 = a_j^0 - \frac{1}{2}C^2\left(a_{j+1}^i - 2a_j^i + a_{j-1}^i\right)$$

The algorithm is,

(1) Initialize $a_j^0 = I(x_j)$ for $j = 0, \ldots N_x$.

(2) Compute a_j^1 and set $a_j^1 = 0$ for the boundary points $i = 0$ and $i = N_x$, for $i = 1, \ldots N - 1$

(3) For each time level $i = 1, \ldots N_t - 1$

  (a) find a_j^{i+1} for $j = 1, \ldots N_x - 1$.

  (b) set $a_j^{i+1} = 0$ for the boundary points $j = 0, j = N_x$.

Iterative Simulation Algorithm

The required formulae are,

$$C = c\frac{\Delta t}{\Delta x}.$$

$$a_j^{i+1} = -a_j^{i-1} + 2a_j^i + C^2\left(a_{j+1}^i - 2a_j^i + a_{j-1}^i\right)$$

$$a_j^1 = a_j^0 - \frac{1}{2}C^2\left(a_{j+1}^i - 2a_j^i + a_{j-1}^i\right)$$

The algorithm is,

(1) Initialize $a_j^0 = I(x_j)$ for $j = 0, \ldots N_x$.

(2) Compute a_j^1 and set $a_j^1 = 0$ for the boundary points $i = 0$ and $i = N_x$, for $i = 1, \ldots N - 1$

(3) For each time level $i = 1, \ldots N_t - 1$

  (a) find a_j^{i+1} for $j = 1, \ldots N_x - 1$.

  (b) set $a_j^{i+1} = 0$ for the boundary points $j = 0, j = N_x$.

Iterative Simulation Algorithm

The required formulae are,

$$C = c\frac{\Delta t}{\Delta x}.$$
$$a_j^{i+1} = -a_j^{i-1} + 2a_j^i + C^2\left(a_{j+1}^i - 2a_j^i + a_{j-1}^i\right)$$
$$a_j^1 = a_j^0 - \frac{1}{2}C^2\left(a_{j+1}^i - 2a_j^i + a_{j-1}^i\right)$$

The algorithm is,

(1) Initialize $a_j^0 = I(x_j)$ for $j = 0, \ldots N_x$.

(2) Compute a_j^1 and set $a_j^1 = 0$ for the boundary points $i = 0$ and $i = N_x$, for $i = 1, \ldots N - 1$

(3) For each time level $i = 1, \ldots N_t - 1$
   (a) find a_j^{i+1} for $j = 1, \ldots N_x - 1$.
   (b) set $a_j^{i+1} = 0$ for the boundary points $j = 0, j = N_x$.

Iterative Simulation Algorithm

The required formulae are,

see previous mesh drawing

$$C = c\frac{\Delta t}{\Delta x}.$$

t i+1

$$a_j^{i+1} = -a_j^{i-1} + 2a_j^i + C^2\left(a_{j+1}^i - 2a_j^i + a_{j-1}^i\right)$$

$$a_j^1 = a_j^0 + \frac{1}{2}C^2\left(a_{j+1}^i - 2a_j^i + a_{j-1}^i\right)$$

formulae formed through reorganisation of discrete wave equn!

The algorithm is,

(1) Initialize $a_j^0 = I(x_j)$ for $j = 0, \ldots N_x$.

(2) Compute a_j^1 and set $a_j^1 = 0$ for the boundary points $i = 0$ and $i = N_x$, for $i = 1, \ldots N - 1$

(3) For each time level $i = 1, \ldots N_t - 1$

   (a) find a_j^{i+1} for $j = 1, \ldots N_x - 1$.

   (b) set $a_j^{i+1} = 0$ for the boundary points $j = 0, j = N_x$.