Application of Newton's Method Shiny App Dashboard

Matthew Paul, Abdel Shehata, Alan Wang
STA 323 Team 1

Project Motivation

- We learned the following in STA 323
  - Newton’s method for root finding
  - How to create Shiny apps
- Combine what we learned to create interactive Shiny app dashboard
  - Demonstrates how Newton’s method approximates roots of a univariate function
  - Explore further applications of Newton’s Method in other fields

Shiny App Dashboard Overview

Page 1: Root Finder

Page 2: Damped Driven Harmonic Oscillator

Page 3: Internal Rate of Return (IRR) Calculator

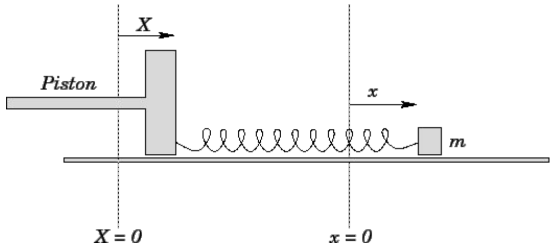
Page 1: Newton's Method Root Finder

- Calculate the root of a univariate function f(x)
- We use the following standard procedural update as discussed in class:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

Page 2: Damped Driven Harmonic Oscillator

- Simulates a damped driven harmonic oscillator using Newton's Method for numerical integration
- Can be expressed using the following second-order ordinary differential equation shown below:



$$m\frac{d^2x}{dt^2} + c\frac{dx}{dt} + kx = F_0 \sin(\omega t)$$

- m = oscillator mass
- x = oscillator displacement
- c = damping coefficient
- dx/dt = oscillator velocity
- k = spring constant
- F_0 = force amplitude
- ω = angular frequency
- t = time

Page 2: Damped Driven Harmonic Oscillator

Apply Newton's Method to the damped driven harmonic oscillator's ODE

- Discretize the ODE using a time step (dt) and then apply the method to update the displacement (x) and velocity (v) at each time step
- Second derivative (left) and first derivative (right) approximations

$$\frac{d^2x}{dt^2} \approx \frac{x_{n+1} - 2x_n + x_{n-1}}{dt^2} \qquad \frac{dx}{dt} \approx \frac{x_{n+1} - x_{n-1}}{2dt}$$

- Substitute into continuous ODE:

$$m\frac{x_{n+1} - 2x_n + x_{n-1}}{dt^2} + c\frac{x_{n+1} - x_{n-1}}{2dt} + kx_n = F_0 \sin(\omega t_n)$$

- Multiply by 2dt^2 to get rid of denominators (left) and rewrite (right):

$$2m(dt)(x_{n+1} - 2x_n + x_{n-1}) + c(x_{n+1} - x_{n-1}) + 2k(dt^2)x_n = 2F_0(dt^2)\sin(\omega t_n)$$

$$x_{n+1} - x_n - dt * v_n - 0.5 * dt^2 * (F_0 * \sin(\omega t_n) - c * v_n - k * x_{n+1})/m = 0$$

Page 2: Damped Driven Harmonic Oscillator

- Rearrange:

$$f(x_{n+1}) = x_{n+1} - x_n - dt * v_n - 0.5 * dt^2 * \frac{(F_0 * \sin(\omega t_n) - c * v_n - k * x_{n+1})}{m}$$

- Take the derivative:

$$f'(x_{n+1}) = \frac{d}{dx_{n+1}} f(x_{n+1}) = 1 - 0.5 * dt^2 * -\frac{k}{m}$$

- Resulting update equation for v:

$$v_{n+1} = v_n + 0.5 * dt * \left(\frac{F_0 * \sin(\omega t_n) - c * v_n - k * x_n}{m} + \frac{F_0 * \sin(\omega t_{n+1}) - c * v_n - k * x_{n+1}}{m} \right)$$

Page 3: Internal Rate of Return (IRR) Calculator

- IRR = the discount rate at which the Net Present Value (NPV) of a series of cash flows becomes zero
- NPV represented using the following equation:

$$NPV = \sum_{i=0}^{n-1} \frac{CF_i}{(1+IRR)^i} = 0$$

- NPV = Net Present Value
- IRR = Internal Rate of Return
- CF_i = cash flow at time i
- n = # of cash flows

Page 3: Internal Rate of Return (IRR) Calculator

- Apply Newton's Method to NPV equation to find IRR
- IRR function is:

$$f(IRR) = \sum_{i=0}^{n-1} \frac{CF_i}{(1+IRR)^i}$$

- Take the derivative:

$$f'(IRR) = -\sum_{i=0}^{n-1} \frac{i \cdot CF_i}{(1+IRR)^{i+1}}$$

Further Improvements

If we continued working on this app, we can improve it in the following ways:

- Include animations of simulations/updates
- Multivariate Newton’s method for root finder
- Add a fallback method to the IRR calculator to make sure it always converges to a percentage

App Demo

Thank you for listening!