Toán kinh tế: MATLAB buổi 2/15

1 Phương pháp chia đôi

```
f = @(x) x^3 + 2*x - 1  % khai báo hàm số f(x) = x^3 + 2x - 1
f(0)                    % → -1
f(2)                    % → 11
```

Mã 1: Phương pháp chia đôi - Module (a)

```
a = 0;
b = 2;
for n = 1:5
    c = (a+b) / 2;
    if f(c) == 0
        c
        break
    elseif f(a) * f(c) < 0
        b = c;
    else
        a = c;
    end
    [a, b]  % a_n, b_n
end
```

Mã 2: Phương pháp chia đôi - Module (b)

```
a = 0;
b = 2;
for n = 1:5
    c = (a+b) / 2;
    if f(c) == 0
        c
        break
    elseif f(a) * f(c) < 0
        b = c;
    else
        a = c;
    end
    err = b - a
    [a, b, err]  % a_n, b_n, ε_n
end
```

Mã 3: Phương pháp chia đôi - Module (b, c)

```
a = 0;
b = 2;
while b - a > 10^-2
    c = (a+b) / 2;
    if f(c) == 0
        c
        break
    elseif f(a) * f(c) < 0
        b = c;
    else
        a = c;
```

```
    end
    err = b - a
    [a, b, err]   % a_n, b_n, ε_n
end
```

Mã 4: Phương pháp chia đôi - Module (d)

```
log2((2-0) / 10^-6)
```

Mã 5: Phương pháp chia đôi - Module (e)

2 Phương pháp Newton

```
f = @(x) x^3 - x^2 - 3

% Cách 1
syms x
fplot(f(x), [1, 4])

% Cách 2
diff(f(x), 2)
subs(diff(f(x)), 1)
f(1), f(4)
```

Mã 6: Phương pháp Newton - Module (a)

```
syms t
df = @(x) subs(diff(f(t)), x)
df(x)

x = zeros(1, 4)
x(1) = 4
for n = 1:3
    x(n+1) = x(n) - f(x(n)) / df(x(n))
end
```

Mã 7: Phương pháp Newton - Module (b)

```
M = 22
m = min(abs(df(1)), abs(df(4)))

e = zeros(1, 4)
for n = 2:4
    e(n) = M / 2 / m * (x(n) - x(n-1))^2
end
```

Mã 8: Phương pháp Newton - Module (c)

```
x0 = 4
while true
    x = vpa(x0 - f(x0) / df(x0))
    e = vpa(M / 2 / m * (x - x0)^2)
    x0 = x;
    if e < 10^-6
        break
    end
```

```
end
```

Mã 9: Phương pháp Newton - Module (d)

```
x = zeros(1, 4)
x(1) = 4
for n = 1:3
    x(n+1) = x(n) - f(x(n)) / df(4)
end
```

Mã 10: Phương pháp Newton - Module (e)

```
% Cách 1
X = linspace(1, 4, 101)
d2f = @(x) -abs(subs(diff(f(t), 2), x))
Y = abs(vpa(d2f(X)))
M = max(Y)

% Cách 2
g = @(x) -abs(subs(diff(f(t), 2), x))
[xmin, M] = fminbnd(g, 1, 4)
vpa(-M)
```

Mã 11: Phương pháp Newton: tìm M dưới góc độ thực hành

3 Phương pháp lặp điểm bất động

```
g = @(x) nthroot(x^2 + 3, 3)

% Cách 1
syms x
diff(g(x))

g(1), g(4)

simplify( diff(g(x), 2) )

syms t
dg = @(x) subs(diff(g(t)), t, x)

for x = [1, 4, 3]
    vpa(abs(dg(x)))
end

q = vpa(abs(dg(3)))


% Cách 2
syms x
fplot(g(x), [1, 4])
fplot(abs(dg(x)), [1, 4])

X = linspace(1, 4, 101)
Y = abs(vpa(dg(X)))
q = max(Y)
```

Mã 12: Phương pháp lặp điểm bất động - Module (a)

```
x = zeros(1, 4);
x(1) = 2.5;
for n = 1:3
    x(n+1) = g(x(n));
end

x
```

Mã 13: Phương pháp lặp điểm bất động - Module (b)

```
e = zeros(1, 4);
for n = 2:4
    e(n) = q / (1-q) * abs(x(n) - x(n-1));
end

e
```

Mã 14: Phương pháp lặp điểm bất động - Module (c)

```
x0 = 2.5;
while true
    x = g(x0)
    e = q / (1-q) * abs(x - x0)
    x0 = x;
    if e < 10^-4
        break
    end
end
```

Mã 15: Phương pháp lặp điểm bất động - Module (d)

```
x0 = 2.5
x1 = g(x0)
log(10^-10 * (1-q) / abs(x1 - x0)) / log(q)
```

Mã 16: Phương pháp lặp điểm bất động - Module (e)