Nonlinear Rootfinding (Introduction)

Introduction

Problem Statement

If f is linear, $f(x) = mx$. (trivial)

If f is affine, $f(x) = mx + b$. (trivial)

Rootfinding Problem

Given a continuous scalar function of a scalar variable, find a real number r such that $f(r) = 0$.

f

- r is a **root** of the function f.
- The formulation $f(x) = 0$ is general enough; *e.g.*, to solve $g(x) = h(x)$, set $f = g - h$ and find a root of f.

Iterative Methods

Square/overdetermined linear problems.

- Unlike the earlier linear problems, the root cannot be produced in a finite number of operations.
- Rather, a sequence of approximations that formally converge to the root is pursued.

Iteration Strategy for Rootfinding. To find the root of f:

1. Start with an initial iterate, say x_0.
2. Generate a sequence of iterates $x_1, x_2, \ldots$ using an *iteration algorithm* of the form
$$x_{k+1} = g(x_k), \quad k = 0, 1, \ldots$$
3. Continue the iteration process until you find an x_i such that $f(x_i) = 0$. (In practice, continue until some member of the sequence seems to be "good enough".)

MATLAB's FZERO

`fzero` is MATLAB's general purpose rootfinding tool.

Syntax:

```
x_zero = fzero( <function>, <initial iterate> )
x_zero = fzero( <function>, <initial interval> )
[x_zero, fx_zero] = ....
```

numerically found root x_{zero}

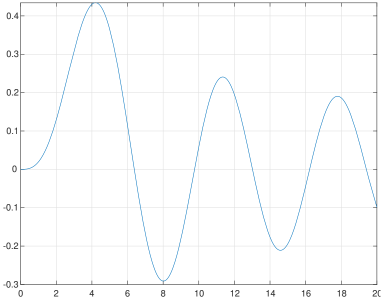
$f(x_{zero})$

Example

The roots of J_m, a Bessel function of the first kind, is found by

- Plot the function.
- Find approximate locations of roots.

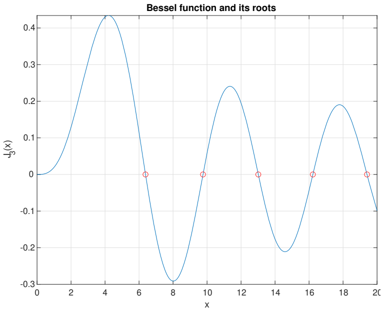
```
J3 = @(x) besselj(3,x);
fplot(J3,[0 20])
grid on
guess = [6,10,13,16,19];
```



Example (cont')

- Then use `fzero` to locate the roots:

```
omega = zeros(size(guess));
for j = 1:length(guess)
  omega(j) = fzero(J3,guess(j));
end
hold on
plot(omega,J3(omega),'ro')
```



Conditioning

- Sensitivity of the rootfinding problem can be measured in terms of the condition number:

$$\text{(absolute condition number)} = \frac{|\text{abs. error in output}|}{|\text{abs. error in input}|},$$

  where, in the context of finding roots of f,

  - input: f (function)
  - output: r (root)

- Denote the changes by:
  - error/change in input: ϵg, where $\epsilon > 0$ is small $\quad (f \mapsto f + \epsilon g)$
  - error/change in output: Δr $\quad (r \mapsto r + \Delta r)$

Conditioning (cont')

- The *perturbed equation*

$$f(r + \Delta r) + \epsilon g(r + \Delta r) = 0$$

  is linearized to (Taylor expansion)

$$f(r) + f'(r)\Delta r + g(r)\epsilon + g'(r)\epsilon\Delta r \approx 0,$$

  ignoring $O((\Delta r)^2)$ terms[1].

- Since $f(r) = 0$, we solve for Δr to get

$$\Delta r \approx -\epsilon \frac{g(r)}{f'(r) + \epsilon g'(r)} \approx -\epsilon \frac{g(r)}{f'(r)},$$

  for small ϵ compared with $f'(r)$.

[1] That is, terms involving $(\Delta r)^2$ and higher powers of Δr

Conditioning (cont')

- Therefore, the absolute condition number of the rootfinding problem is

$$\kappa_{f \mapsto r} = \frac{1}{\left|f'(r)\right|},$$

which implies that the problem is highly sensitive whenever $f'(r) \approx 0$.

- In other words, if $|f'|$ is small at the root, a computed *root estimate* may involve large errors.

Residual and Backward Error

- Without knowing the exact root, we cannot compute the error.
- But the **residual** of a root estimate $\tilde{r}$ can be computed:

$$\text{(residual)} = f(\tilde{r}).$$

- Small residual *might* be associated with a small error.
- The residual $f(\tilde{r})$ is the *backward error* of the estimate.

Multiple Roots

Definition 1 (Multiplicity of Roots)

Assume that r is a root of the differentiable function f. Then if

$$0 = f(r) = f'(r) = \cdots = f^{(m-1)}(r) \quad \text{but} \quad f^{(m)}(r) \neq 0,$$

we say that f has a root of **multiplicity** m at r.

- We say that f has a **multiple root** at r if the multiplicity is greater than 1.
- A root is called **simple** if its multiplicity is 1.
- If r is a multiple root, the condition number is infinite.
- Even if r is a simple root, we expect difficulty in numerical computation if $f'(r) \approx 0$.