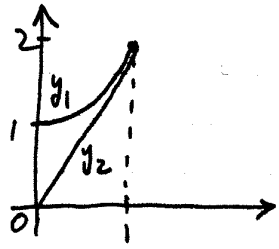
§7. Shooting method for Boundary Value Problems.

(a) <u>Introduction</u>: initial value problems (IVP's) vs. boundary value problems (BVP's). Latter much more difficult; can't march with RK4.

Concrete example: $\begin{cases} y_1' = y_2 & y_1(0)=1 \\ y_2' = \dfrac{2y_1}{x^2+1} & y_2(0)=0. \end{cases}$

Solution: $y_1(x) = x^2+1$, $y_2(x) = 2x$

- Can be solved by RK4, maybe with variable step size h.



* What if BVP with: $\begin{cases} y_1(0)=1 \\ y_2(1)=2 \end{cases}$?

Strategy: (a) Recast into IVP by <u>guessing</u> the missing initial condition, say $y_2(0)=0.2$.

(b) March by RK4 to the end, and see if $y_2(1)$ matches the boundary value 2.

(c) If not, adjust guess of $y_2(0)$ to match $y_2(1)$ to target. But how?

- Bisecting? Now looks like root-finding!
- Newton-Raphson method, with numerical differentiation:

* Discrepancy $F = y_2(1) - 2$ is a function of guessed $y_2(0) = x$. Then find x so $F(x) = 0$! But functional form not explicitly given $\Longrightarrow$ Numerical differentiation for $F'(x)$.

General form: $\begin{cases} \dfrac{dy_i(x)}{dx} = g_i(x, y_1, \cdots, y_N), \quad i=1,2,\cdots,N \\ y_j(x_1) = y_{j1}, \quad j=1,\cdots,n_1 \\ y_k(x_2) = y_{k2}, \quad k=n_1+1,\cdots,N, \quad n_2 \end{cases}$

over $[x_1, x_2]$.

$(n_2 = N - n_1)$

(b) Shooting to the other boundary. $[x_1, x_2]$.

(i) Expand boundary condition at x_1 to form complete I.V.P.:

$$y_k(x_1) = V_k, \quad k = n_1+1, \cdots, N.$$

(ii) March to x_2 to get $y_i(x_2)$, $i = 1, \cdots, N$

(iii) Formulate discrepancy vector:

$$F_k = y_k(x_2) - y_{k2}, \quad k = n_1+1, \cdots, N.$$

as implicitly dependent on V_k values.

(iv) Newton's method for root-finding:

$$F_k(V_k) = 0 \;:$$

$$\underline{\underline{J}} \cdot \delta \underline{V} = -\underline{F}, \quad n_2 \times n_2 \text{ system.}$$

$$J_{ij} = \frac{\partial F_i}{\partial V_j}, \quad i, j = 1, 2, \cdots, n_2.$$

(shifted from $n_1+1, \cdots, N$)

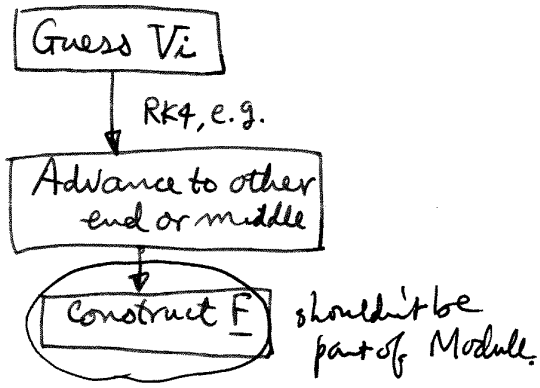
$$\frac{\partial F_i}{\partial V_j} = \frac{F_i(V_1, \cdots, V_j + \Delta V_j, \cdots, V_{n_2}) - F_i(V_1, \cdots, V_j, \cdots, V_{n_2})}{\Delta V_j}$$

Now we can draw the flow-chart of the shooting algorithm.

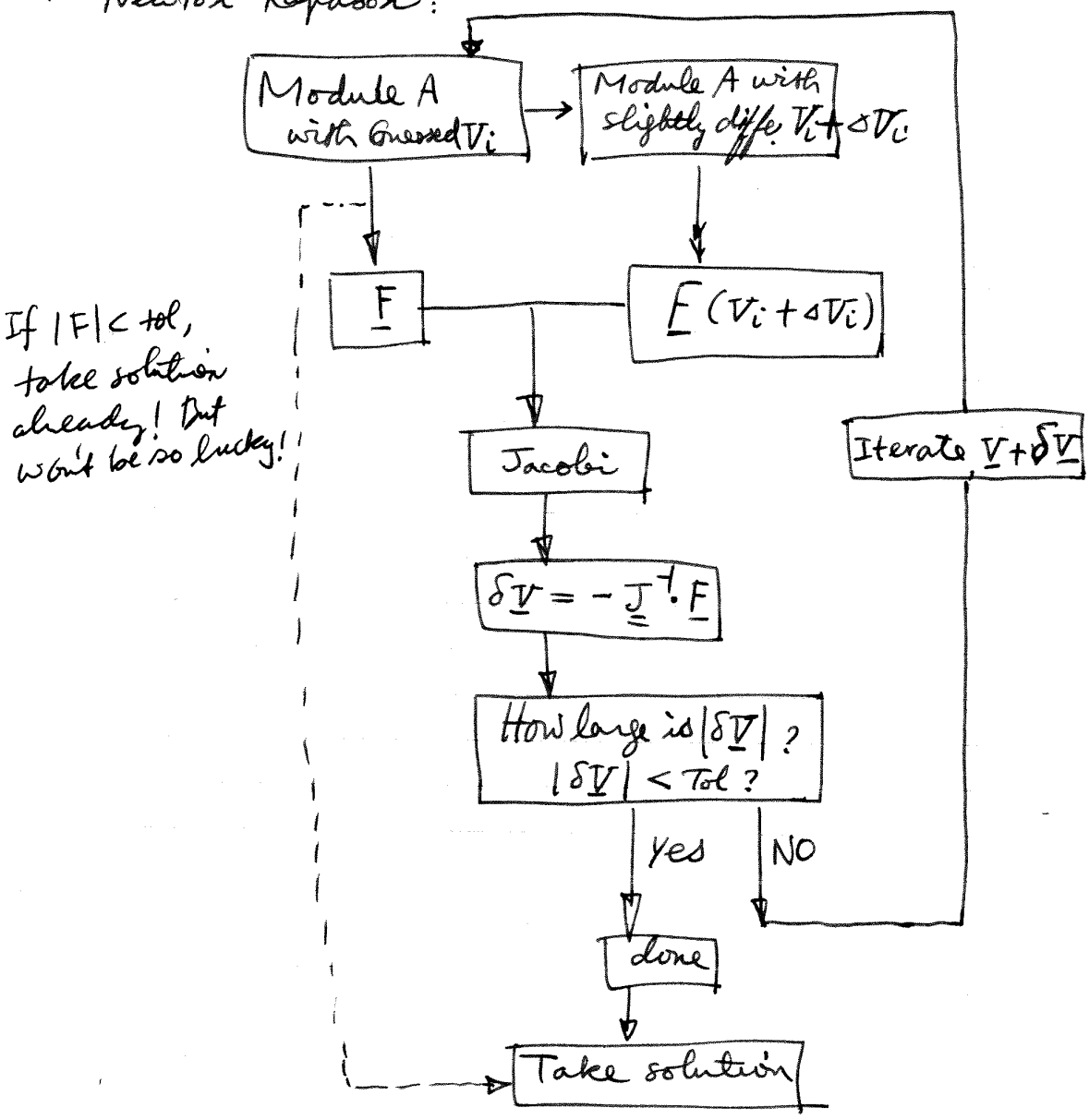
* Could guess missing BC at x_2, and march back toward x_1. Depends on which end has fewer "missing BC's".

Flowchart for shooting scheme:

* Module A:



* Newton-Raphson:



(c) More general expressions of BC's.

(i) BC's at x_1, x_2 needn't be neatly divided between to groups of y, $j=1,\cdots,n_1$, $k=n_1+1,\cdots,N$.

For example: $(N=4)$

x_1	x_2
$y_1 = a$	$y_1 = c$
$y_2 = b$	$y_2 = d$
Guess $y_3 = V_1$	$\begin{cases} F_1 = y_1(x_2) - c \\ F_2 = y_2(x_2) - d \end{cases}$
$y_4 = V_2$	Shoot

$\underline{2\times 2 \text{ Newton's}}$

(ii) BC's may be algebraic expressions. Maybe more convenient to guess values of these instead of y's:

x_1	x_2
$y_1 + \sqrt{y_4^2 + y_3^2} = 2$	$y_2 + y_3 + e^{y_4} = 10$
$y_4 - \sin\sqrt{y_4^2 + y_3^2} = 1$	$y_3^2 + y_4^2 = 1$

* If guessing $y_1 = V_1$, $y_2 = V_2$

Hard to compute y_3, y_4.

* Guess $\sqrt{y_4^2 + y_3^2} = V_2$, $y_2 = V_1$

then $y_1 = 2 - V_2$, $y_4 = 1 + \sin V_2$, $y_3 = [V_2^2 - (1+\sin V_2)^2]^{1/2}$

$\Rightarrow$ March

$$\begin{cases} F_1 = y_2 + y_3 + e^{y_4} - 10 \\ F_2 = y_3^2 + y_4^2 - 1 \end{cases}$$

(iii) In general: given $B_{1j}(x_1, y_1, \cdots, y_N) = 0$, $j = 1,\cdots,n_1$

$B_{2k}(x_2, y_1, \cdots y_N) = 0$, $k = 1,\cdots,n_2$.

Guess n_2 values at x_1 so as to get IVP:

$$y_i(x_1) = y_i(x_1, V_1, \cdots, V_{n_2}), \quad i = 1,\cdots,N.$$

Discrepancy vector:

$$F_k = B_{2k}(x_2, y_1, \cdots, y_N), \quad k = 1,\cdots,n_2.$$

(d) Shooting to intermediate point.

- Initial guess so bad that marching blows up before reaching the other end of domain;
- Integral singularities at ends that allow one to march out of, but not integrate into (see Numerical Recipe, p.751 for more discussion).

$\Rightarrow$ March from both ends to meet at some intermediate point.

(i) Expand both ends into "full" BC's.

$$y_j(x_1) = y_j(x_1, U_1, \cdots, U_{n_2}), \; j=1,\cdots,N$$

$$y_k(x_2) = y_k(x_2, V_1, \cdots, V_{n_1}), \; k=1,\cdots,N$$

(Total guessed values: $U_1, \cdots, U_{n_2}, V_1, \cdots, V_{n_1}$) Numbering N.

(ii) March both IVP's toward middle x_m;

(iii) Construct N-dimensional discrepency vector at x_m:

$$F_i = y_i(x_m, U_1, \cdots, U_{n_2}) - y_i(x_m, V_1, \cdots, V_{n_1}).$$

(iv) Root-finding for N "nonlinear equations":

$$F_i(U_1, \cdots, U_{n_2}, V_1, \cdots, V_{n_1}) = 0$$

by Newton's method. $\underline{\underline{J}}$: $N \times N$, larger system than if shooting from one end.

§8. Finite difference for B.V.P.'s of ODE's

[Simplified notes to avoid complex index notation; ref: Num. Recipes §17.3]

A coupled set of N equations, set on $x \in [x_a, x_b]$:

$$\begin{cases} \dfrac{dy_i}{dx} = g_i(x, y_1, y_2, \cdots, y_N), \quad i = 1, 2, \cdots, N \\ B_j(x_a, y_1(x_a), y_2(x_a), \cdots, y_N(x_a)) = 0, \quad j = 1, \cdots, n_1 \\ C_K(x_b, y_1(x_b), y_2(x_b), \cdots, y_N(x_b)) = 0, \quad K = 1, 2, \cdots, n_2 \end{cases}$$

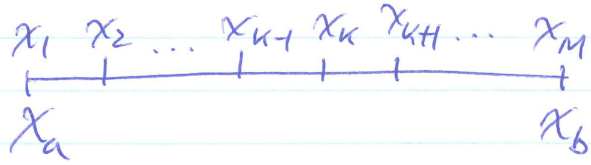
$$n_1 + n_2 = N$$

Shorthand in vectorial form:

$$\frac{d\underline{y}}{dx} = \underline{g}(x, \underline{y})$$

$$\underline{B}(x_a, \underline{y}(x_a)) = 0, \quad \underline{C}(x_b, \underline{y}(x_b)) = 0$$

(a) Discretization: $x_1 = x_a, x_2, \cdots, x_M = x_b$.



$$\frac{\underline{y}_K - \underline{y}_{K-1}}{x_K - x_{K-1}} = \underline{g}(x, \underline{y})$$

what values to take?

Now equations are coupled anyway, no need to preserve explicitness. We may as well use central differencing for 2nd-order accuracy:

$$\underline{y}_K = \underline{y}_{K-1} + (x_K - x_{K-1})\, \underline{g}\left(\frac{x_{K-1} + x_K}{2}, \frac{\underline{y}_{K-1} + \underline{y}_K}{2}\right)$$

For $M-1$ intervals, a total of $(M-1) \times N$ equations

Add BC's: $\left.\begin{array}{l} \underline{B}(x_1, \underline{y}(x_1)) = 0 \\ \underline{C}(x_M, \underline{y}(x_M)) = 0 \end{array}\right\}$ N equations

$\Rightarrow$ $M \times N$ equations for $M \times N$ unknowns.

(b) Newton's Method for solving the nonlinear system.

* How to order unknowns? Finite difference concerns only 2 neighbors ($k-1$ & k). So order $\underline{y}$ at each point and then aggregate:

$$\underline{y}_k = \begin{bmatrix} y_{1,k} \\ y_{2,k} \\ \vdots \\ y_{N,k} \end{bmatrix}, \qquad \underline{Y} = \begin{bmatrix} \underline{y}_{k=1} \\ \underline{y}_2 \\ \vdots \\ \underline{y}_M \end{bmatrix}$$

Details of Jacobian matrix given in N.R. book. For illustration of block diagonal structure, consider 5 variables on 4 mesh points, with 3 BC's at x_1, 2 BC's at x_M.

* Internal block: N rows for N equations for $y_1, y_2, \cdots, y_N$
  2N columns as $\underline{y}_{k-1}, \underline{y}_k$

* Partial blocks at top & bottom:

$$\begin{array}{l} \text{N eqns for N } y \text{ components at } k. \left\{\begin{array}{l} y_{1,k} \rightarrow \\ y_{2,k} \rightarrow \\ y_{3,k} \rightarrow \\ y_{4,k} \rightarrow \\ y_{5,k} \rightarrow \end{array}\right. \end{array} \quad \underbrace{\begin{array}{ccccc} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \end{array}}_{\underline{y}_{k-1}:\ 5 \text{ components}} \quad \underbrace{\begin{array}{ccccc} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \end{array}}_{\underline{y}_k:\ 5 \text{ components}}$$

$$\underbrace{\qquad\qquad\qquad\qquad}_{2N \text{ columns.}}$$

```
X X X X X                                   V   B
X X X X X                                   V   B
X X X X X                                   V   B
X X X X X X X X X X                         V   B
X X X X X X X X X X                         V   B
X X X X X X X X X X                         V   B
X X X X X X X X X X                         V   B
X X X X X X X X X X                         V   B
          X X X X X X X X X X               V   B
          X X X X X X X X X X               V   B
          X X X X X X X X X X               V   B
          X X X X X X X X X X               V   B
          X X X X X X X X X X               V   B
                    X X X X X X X X X X     V   B
                    X X X X X X X X X X     V   B
                    X X X X X X X X X X     V   B
                    X X X X X X X X X X     V   B
                    X X X X X X X X X X     V   B
                              X X X X X     V   B
                              X X X X X     V   B
```

Figure 17.3.1. Matrix structure of a set of linear finite-difference equations (FDEs) with boundary conditions imposed at both endpoints. Here X represents a coefficient of the FDEs, V represents a component of the unknown solution vector, and B is a component of the known right-hand side. Empty spaces represent zeros. The matrix equation is to be solved by a special form of Gaussian elimination. (See text for details.)

```
1     X X                                   V   B
  1   X X                                   V   B
    1 X X                                   V   B
      1       X X                           V   B
        1     X X                           V   B
          1   X X                           V   B
            1 X X                           V   B
              1 X X                         V   B
                1       X X                 V   B
                  1     X X                 V   B
                    1   X X                 V   B
                      1 X X                 V   B
                        1       X X         V   B
                          1     X X         V   B
                            1   X X         V   B
                              1 X X         V   B
```