Lecture 8: Roots of Equations – System of Nonlinear Equations

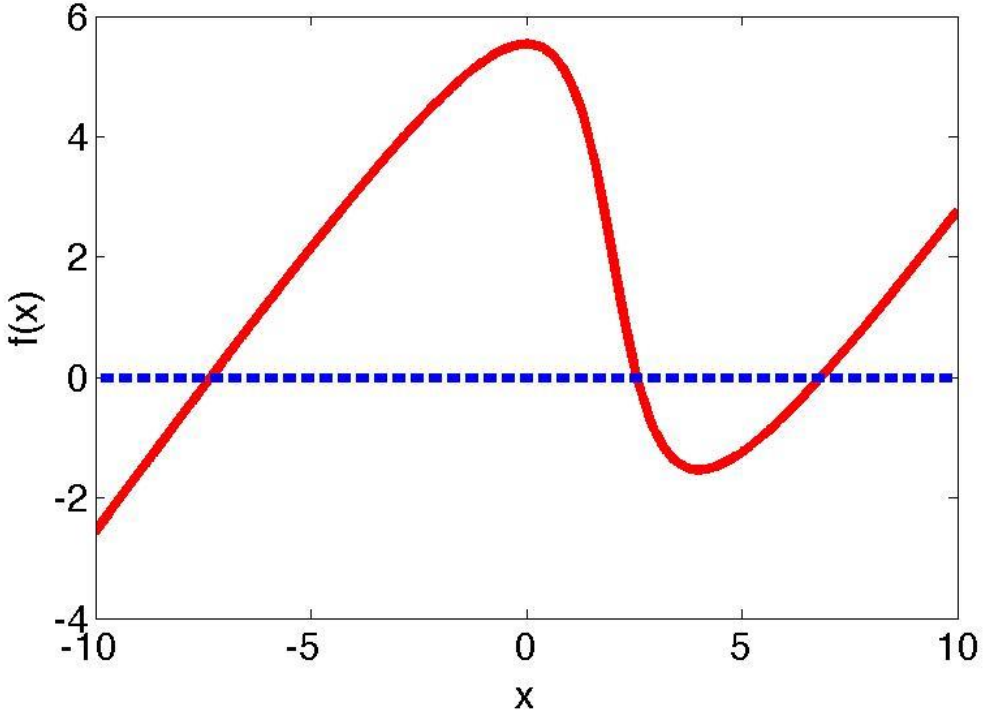
CBE 206

10.8.2019

Review of Open Methods: An Example: 1) Fixed-point Iteration

$$f(x) = x - 5\tan^{-1}(x-2)$$

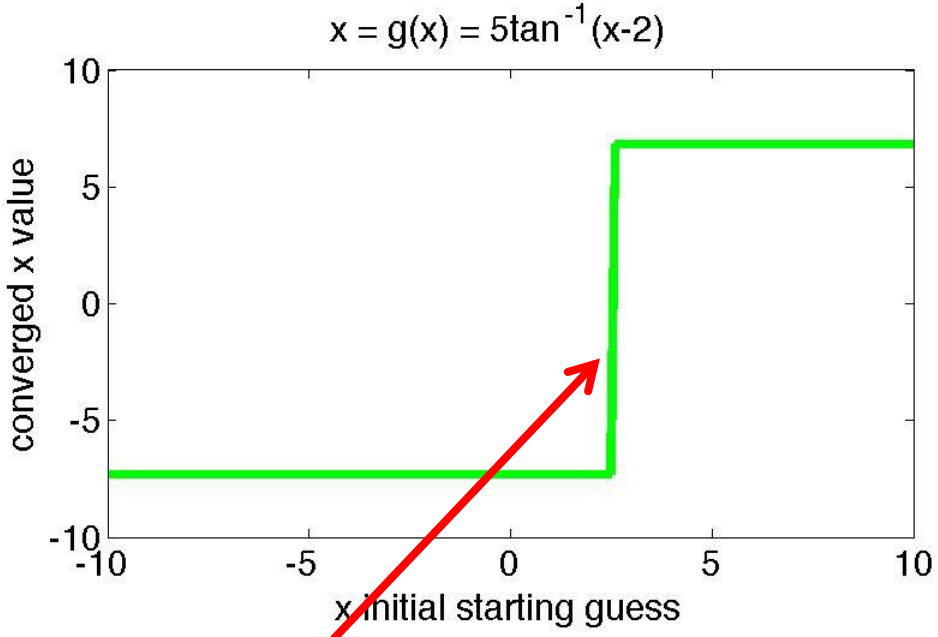
- Plot this equation.
- We see that there are three solutions.



- Next, let us solve this problem using fixed point iteration.
- Let us reformulate this equation into $x = g(x) \rightarrow x = 5\tan^{-1}(x-2)$

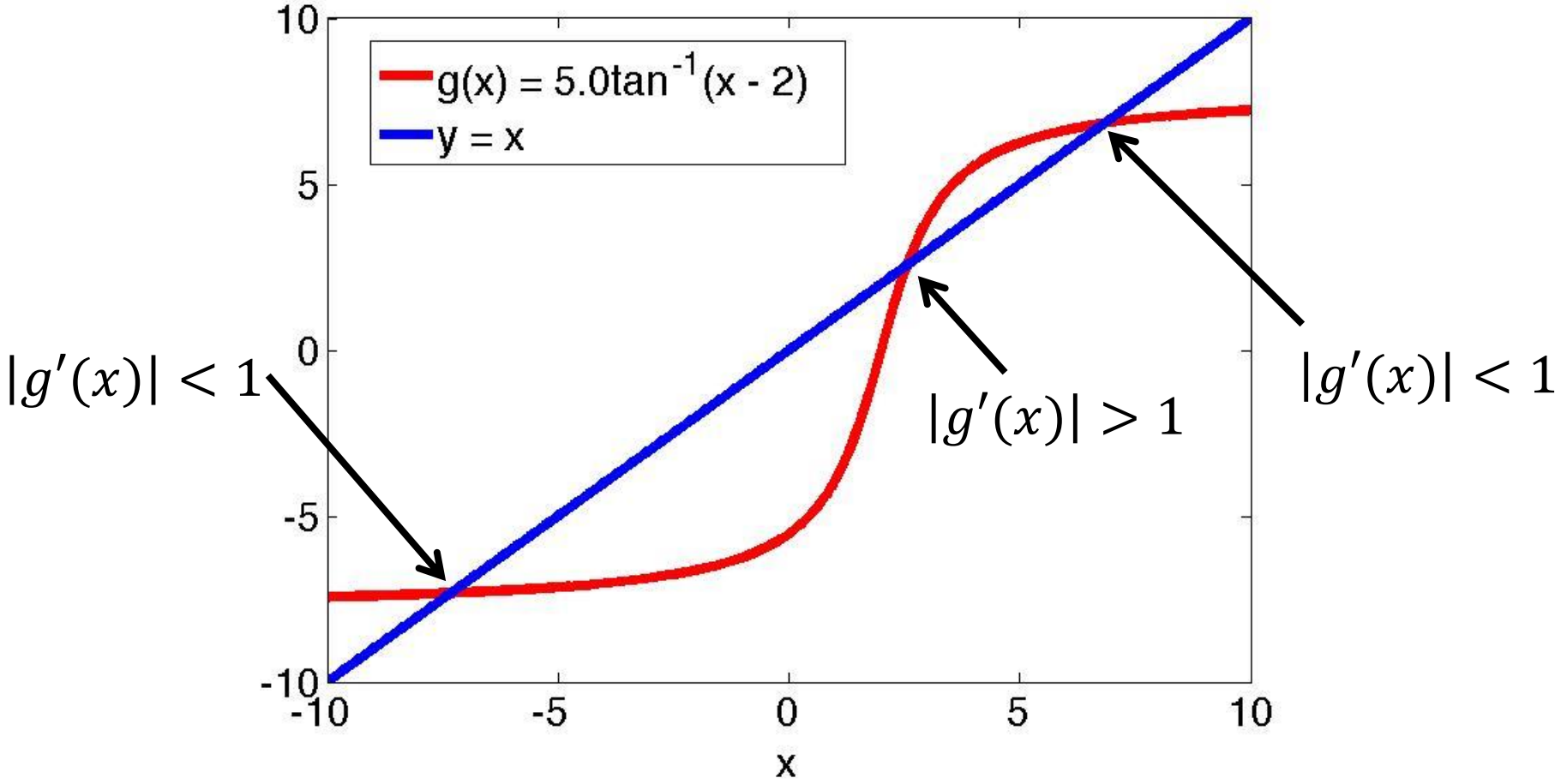
Review of Open Methods: An Example: 1) Fixed-point Iteration with Different Starting Guess

- Simple fixed point iteration method is used with 30 (no error threshold defined) different initial guess for x.
- One can see that we manage to obtain 2 out of the 3 roots ($x = -7.3195$ and $x = 6.8340$).
- Why can't we obtain the 3rd one?

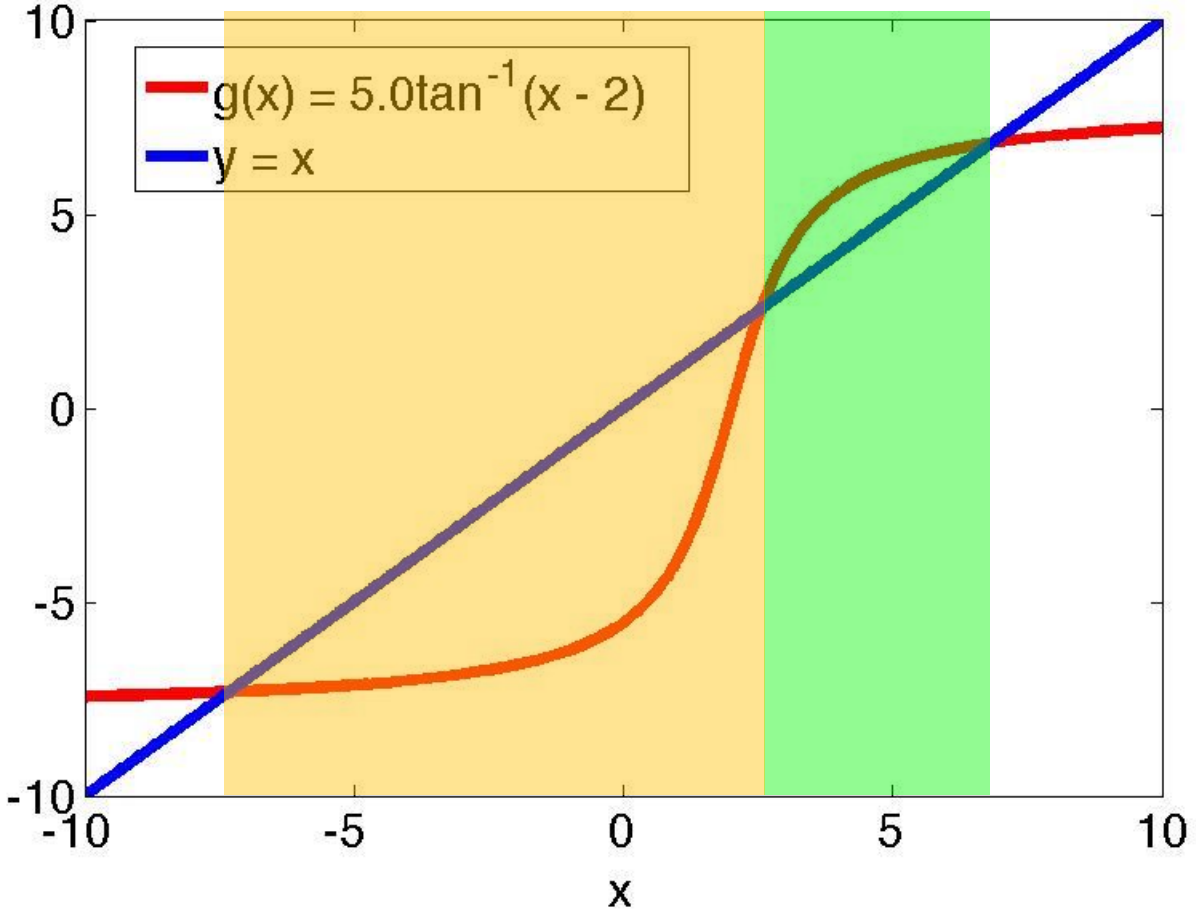


This discontinuity occurs where the 3rd solution is located!

Review of Open Methods: An Example: 1) Fixed-point Iteration x = g(x), slopes



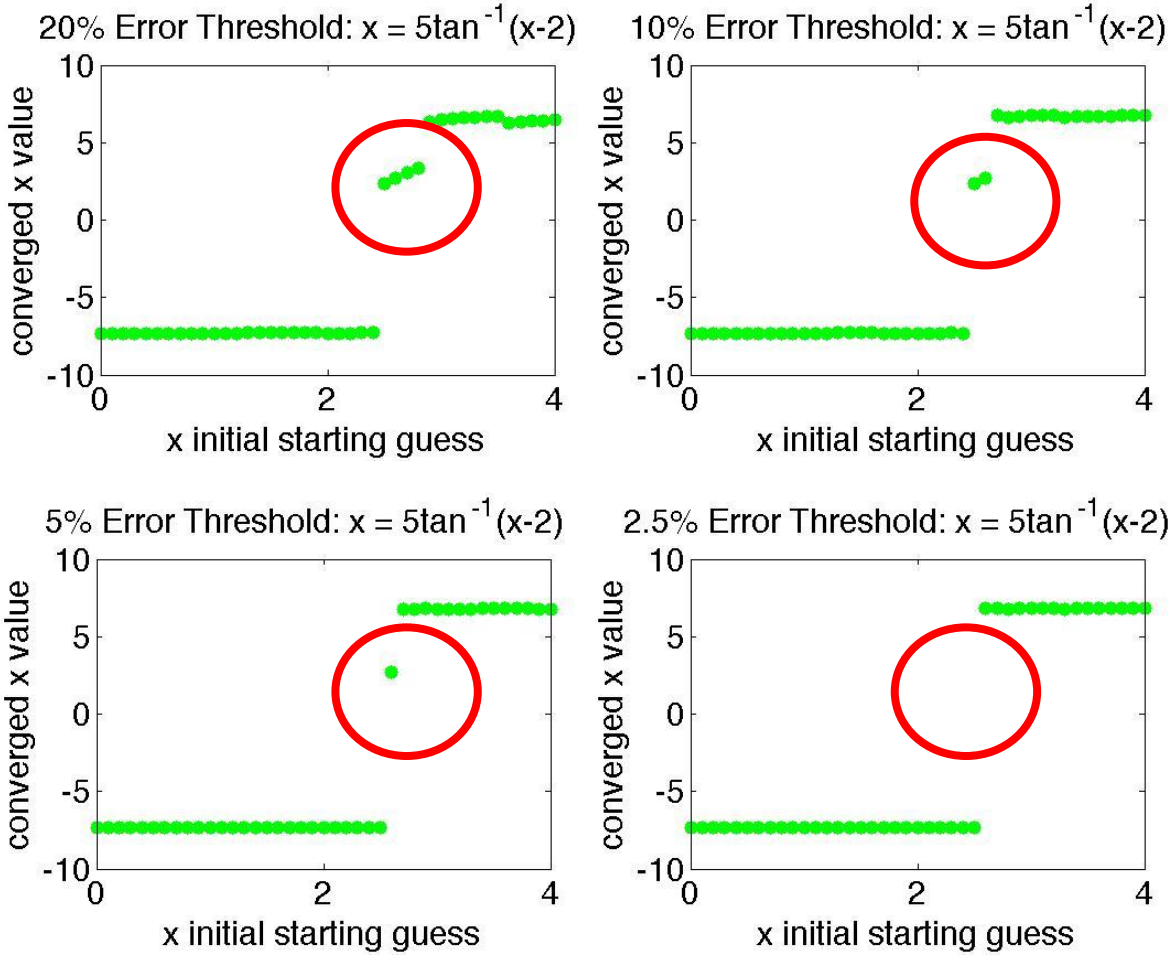
Review of Open Methods: An Example: 1) Fixed-point Iteration x = g(x), Additional Interpretation



- In the green region, g(x) is always larger than x.
- For any x chosen, the next choice of x (i.e. g(x)) is larger than the previous one.
- We move away from the solution at x = 2.5
- Similar argument for the yellow region.

Review of Open Methods: An Example: Repeat Slide 3 with Error Threshold

- In Slide 3, we obtained the convergent behavior for N = 30 iterations, without defining the error thresholds.
- For large error thresholds, we can get to the 3rd root since some of the initial guesses are good enough already.
- As error threshold decreases, the 3rd root disappears from our method.



KAIST

Review of Open Methods: An Example: 2) Newton-Raphson Method

$$f(x) = x - 5\tan^{-1}(x-2)$$

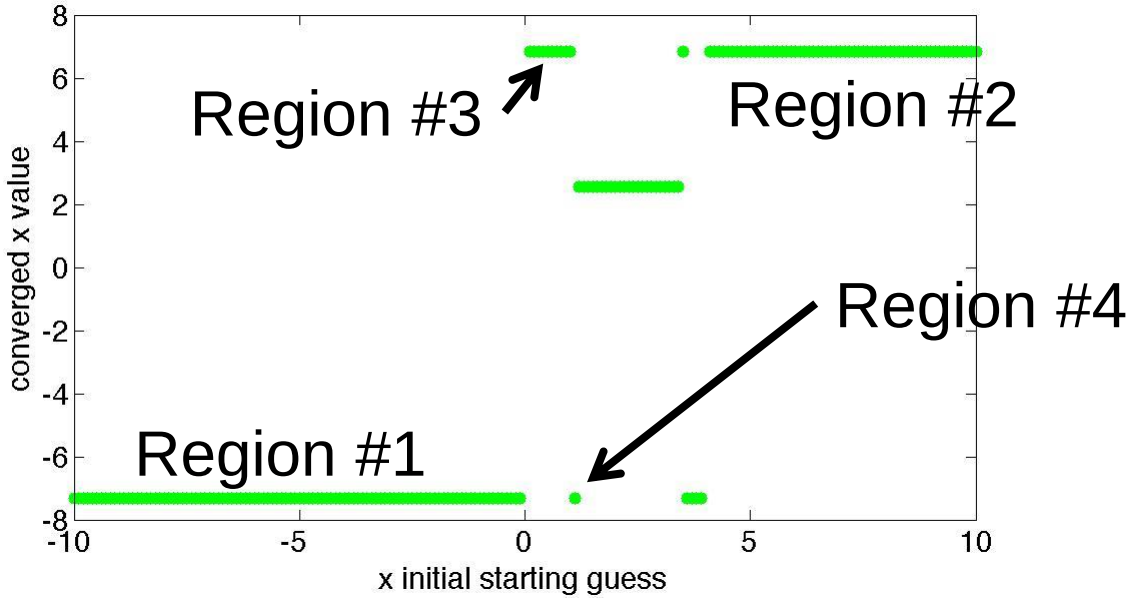
- For Newton-Raphson method, we need to compute the derivative of $f(x)$.

$$f'(x) = 1 - \frac{5}{1+(x-2)^2}$$

$$x_{i+1} = x_i - \frac{x_i - 5\tan^{-1}(x_i-2)}{1 - \frac{5}{1+(x_i-2)^2}}$$

Review of Open Methods: An Example: 2) Newton-Raphson Method with Different Starting Guess

- Newton-Raphson method is used with 30 different starting values of x.
- One can see that we manage to obtain all 3 out of the 3 solutions ($x = -7.3195$, $x = 2.5627$, and $x = 6.8340$).
- The initial starting x values that lead to the three solutions are not trivially predictable.



Review of Open Methods: An Example: 2) Newton-Raphson Method with Different Starting Guess

- At $x = 0$ and $x = 4$, the slope of $f(x)$ becomes zero (See Slide 2). i.e.

$$f'(x) = 1 - \frac{5}{1 + (x-2)^2} \rightarrow \quad f'(x) = 0 \; for \; x = 0 \; and \; 4$$

- With MATLAB (and other software), the Newton-Raphson method breaks down when the slope becomes zero since we are dividing by zero.
- A good algorithm would check to see if $f'(x) = 0$.
- What is happening elsewhere?

Region #1 $-10 < x < 0$: $f'(x) > 0 \rightarrow x_{i+1} > x_i \; when \; f(x) < 0$ and $x_{i+1} < x_i \; when \; f(x) > 0$ (move towards the correct direction for $x = -7.1395$ root).

Region #2 $4 < x < 10$: $f'(x) > 0 \rightarrow x_{i+1} > x_i \; when \; f(x) < 0$ and $x_{i+1} < x_i \; when \; f(x) > 0$ (move towards the correct direction for $x = 6.8340$ root).

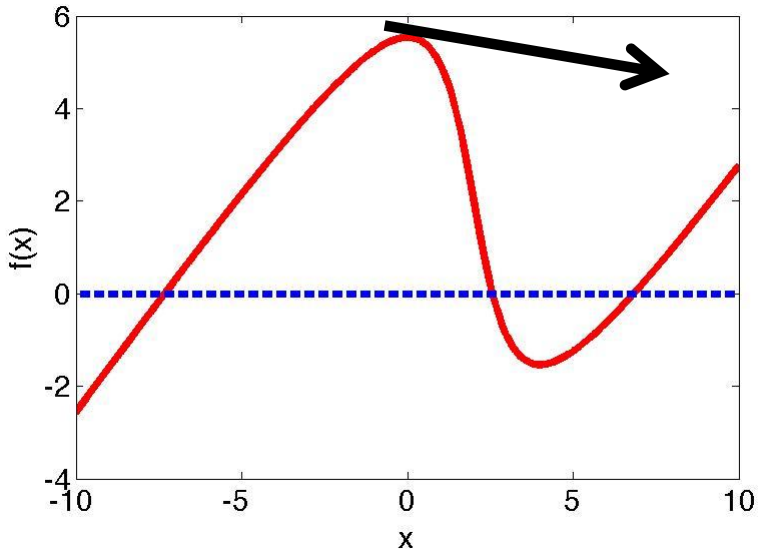
Review of Open Methods: An Example: 2) Newton-Raphson Method with Different Starting Guess

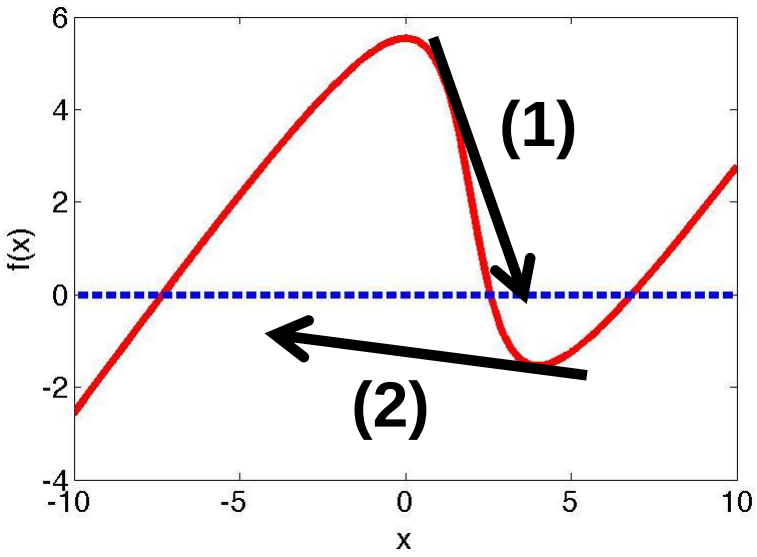
Region #3 $0 < x < 1.05$:

(Overshoot to right!!)

Region #4 $1.05 < x < 1.15$:

(Overshoot to left!!)





And so forth ….

System of Nonlinear Equations

- Thus, far we have solved one nonlinear equation with one unknown (i.e. $f(x) = 0$)
- We can solve a system of nonlinear equations with n equations and n unknowns.

$$f_1(x_1, x_2, \dots x_n) = 0$$
$$f_2(x_1, x_2, \dots x_n) = 0$$
$$\dots\dots$$
$$f_n(x_1, x_2, \dots x_n) = 0$$

- In Lecture 4 (Slide 2), we talked about system of linear equations.
- System of nonlinear equations is any set of equations that cannot be fitted into the form of linear equations.

System of Nonlinear Equations – Example 1

- Solving system of nonlinear equations is much more complicated than solving linear system of equations.
- Most methods use open methods (not bracket methods).
- Let us look at a simple example.

$$x^2 + xy = 10$$

$$y + 3xy^2 = 57$$

- First, check to see if this is linear or nonlinear…
- Fixed point iteration results in reformulating the equation to look like $x = g(x)$ in 1D. Now we need to do this for two different variables.

- The problem is, there are many ways to formulate this.
- Let us try one way.

Example 1: Fixed Point Iteration

$$x^2 + xy = 10 \quad \rightarrow \quad x = \frac{10 - x^2}{y} \quad \rightarrow \quad x_{i+1} = \frac{10 - {x_i}^2}{y_i}$$

$$y + 3xy^2 = 57 \quad \rightarrow \quad y = 57 - 3xy^2 \quad \rightarrow \quad y_{i+1} = 57 - 3x_{i+1}{y_i}^2$$

The correct answer is, $x = 2$ and $y = 3$. Assuming we know this, let us try a guess nearby: $x = 2.5$ and $y = 2.5$.

The approach:

(1) Plug in $x = 2.5$ and $y = 2.5$ in the first equation. Get x_{new}.
(2) Plug in x_{new} and $y = 2.5$ in the second equation. Get y_{new}.
(3) Repeat this process until (hopefully) convergence.

Example 1: Fixed Point Iteration, Divergent Behavior

Iteration number	1 (initial guess)	2	3	4	5
x	2.5	3.0	9.333	0.0162	-0.3915
y	2.5	0.75	41.25	-25.500	820.763

The solution diverges!! What if we plug into the y equation first?

Iteration number	1 (initial guess)	2	3	4	5
y	2.5	10.125	-170.81	4801.8	-144780
x	2.5	0.7407	-0.0542	0.0021	-0.0001

Still diverges..

Example 1: Fixed Point Iteration, Reformulation

$$x^2 + xy = 10 \rightarrow x = \sqrt{10 - xy} \rightarrow x_{i+1} = \sqrt{10 - x_i y_i}$$

$$y + 3xy^2 = 57 \rightarrow y = \sqrt{\frac{57 - y}{3x}} \rightarrow y_{i+1} = \sqrt{\frac{57 - y_i}{3x_{i+1}}}$$

The solution converges!!

Iteration number	1 (initial guess)	2	3	4	5
x	2.5	1.9365	2.0171	1.9945	2.0019
y	2.5	3.0629	2.9855	3.0046	2.9985

It is difficult to predict a priori what is a good formulation that leads to convergence for system of equations. Thus, we will focus mostly on Newton-Raphson method.

Example Problem 1: Newton Raphson Method

$x_{i+1} = x_i - \frac{f(x_i)}{f\prime(x_i)}$ Newton-Raphson for one variable

One can rearrange the equation in following format.

$$0 = f(x_i) + f'(x_i)(x_{i+1} - x_i)$$

In multi-dimensions, we have multiple functions, $f_1, f_2, \ldots f_N$ with multiple variables $x_1, x_2, \ldots x_N$ such that

$$f_1(x_1, x_2, \ldots x_N) = 0$$
$$f_2(x_1, x_2, \ldots x_N) = 0$$
$$\ldots$$
$$f_N(x_1, x_2, \ldots x_N) = 0$$

Example Problem 1: Newton Raphson Method Setup

- We see that there are multiple variables (e.g. $x_1, x_2, \ldots x_N$).
- We use following notation: $x_{j,i} = j^{th} variable\ at\ i^{th} Newton\ Raphson\ iteration$
- We need to use partial derivatives rather than ordinary derivatives here.
- Ordinary derivatives: $\frac{df}{dx}$ vs partial derivatives: $\frac{\partial f}{\partial x}$
- In partial derivatives, the function has several variables..
- Example: $f(x,y,z) \rightarrow \frac{\partial f}{\partial x}\ (y\ and\ z\ are\ constants), \frac{\partial f}{\partial y}\ (x\ and\ z\ are\ constants)$
- So going back to the formulation in the previous slide,

$$0 = f_1(x_{1,i}, x_{2,i}, \ldots x_{N,i}) + \sum_{j=1}^{N} \frac{\partial f_1}{\partial x_j}(x_{j,i+1} - x_{j,i})$$

$$0 = f_2(x_{1,i}, x_{2,i}, \ldots x_{N,i}) + \sum_{j=1}^{N} \frac{\partial f_2}{\partial x_j}(x_{j,i+1} - x_{j,i})$$

Example Problem 1: Newton Raphson Method Setup - Continued

- In 1D Newton Raphson method, we set $f(x_{i+1}) = 0$.
- We do the similar thing with the multi-dimension case here.

$$0 = f_1(x_{1,i}, x_{2,i}, \dots x_{N,i}) + \sum_{j=1}^{N} \frac{\partial f_1}{\partial x_{j,i}} (x_{j,i+1} - x_{j,i})$$

$$\dots$$

- Writing in matrix form..

$$\begin{bmatrix} -f_1(x_{1,i}, x_{2,i}, \dots x_{N,i}) \\ -f_2(x_{1,i}, x_{2,i}, \dots x_{N,i}) \\ \dots \\ -f_N(x_{1,i}, x_{2,i}, \dots x_{N,i}) \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial x_{1,i}} & \frac{\partial f_1}{\partial x_{2,i}} & \dots & \frac{\partial f_1}{\partial x_{N,i}} \\ \dots & \dots & \dots & \dots \\ \frac{\partial f_N}{\partial x_{1,i}} & \frac{\partial f_N}{\partial x_{2,i}} & \dots & \frac{\partial f_N}{\partial x_{N,i}} \end{bmatrix} \begin{bmatrix} x_{1,i+1} - x_{1,i} \\ x_{2,i+1} - x_{2,i} \\ \dots \\ x_{N,i+1} - x_{N,i} \end{bmatrix}$$

Example Problem 1: Newton Raphson Method Setup - Continued

- This is confusing.. Let us look at what each of these terms represent.

Known from previous iteration

$$\begin{bmatrix} -f_1(x_{1,i}, x_{2,i}, \dots x_{N,i}) \\ -f_2(x_{1,i}, x_{2,i}, \dots x_{N,i}) \\ \dots \\ -f_N(x_{1,i}, x_{2,i}, \dots x_{N,i}) \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial x_{1,i}} & \frac{\partial f_1}{\partial x_{2,i}} & \dots & \frac{\partial f_1}{\partial x_{N,i}} \\ \dots & \dots & \dots & \dots \\ \frac{\partial f_N}{\partial x_{1,i}} & \frac{\partial f_N}{\partial x_{2,i}} & \dots & \frac{\partial f_N}{\partial x_{N,i}} \end{bmatrix} \begin{bmatrix} x_{1,i+1} \\ x_{2,i+1} \\ \dots \\ x_{N,i+1} \end{bmatrix} - \begin{bmatrix} x_{1,i} \\ x_{2,i} \\ \dots \\ x_{N,i} \end{bmatrix}$$

N nonlinear functions whose roots we are trying to obtain :
$f_1: equation\ \#1$, $f_2: equation\ \#2$, ...
$x_{1,i}$: 1st variable at ith Newton-Raphson iteration.

NxN matrix of partial derivatives: Jacobian matrix.

Unknown. In a given Newton-Raphson iteration, this is what we are trying to solve.

Example Problem 1: Newton Raphson Method Setup - Continued

- Let us simplify notation.
- Red box from previous slide: $-\bar{f}(\bar{x}_i)$
- Green box from previous slide: $\bar{J}_f(\bar{x}_i)$
- Blue and purple boxes from previous slide: $(\overline{x_{i+1}} - \bar{x}_i)$
- So we have a nonlinear system of equations that looks as follows.

$$-\bar{f}(\bar{x}_i) = \bar{J}_f(\bar{x}_i)(\overline{x_{i+1}} - \bar{x}_i)$$

- We can take the inverse of the Jacobian and solve for $\overline{x_{i+1}}$.
- But in general, we do not like taking inverses..
- So we are going to solve a system of linear equations..
- Let $\bar{s}_i = (\overline{x_{i+1}} - \bar{x}_i)$. First, solve $\bar{J}_f(\bar{x}_i)\bar{s}_i = -\bar{f}(\bar{x}_i)$.
- Then, $\overline{x_{i+1}} = \bar{s}_i + \bar{x}_i$.

Newton-Raphson Algorithm for System of Nonlinear Equations

$\overline{x_1} = initial\ guess\ to\ the\ problem$

Loop i = 1, 2, 3, …

Solve $\bar{J}_f(\overline{x_i})\overline{s_i} = -\bar{f}(\overline{x_i}) \quad for\ \overline{s_i}$

$\overline{x_{i+1}} = \overline{s_i} + \overline{x_i}$

end

- Upon plugging in an initial guess to the equations, everything becomes linearized.
- Thus, we are essentially solving a system of linear equations at every iteration until we get convergence.
- Termination criterion varies, but in many cases, we can take the norm of the differences between $\overline{x_{i+1}}$ and $\overline{x_i}$ to see if it is smaller than some threshold.

Revisit Example from Slide 12: Newton-Raphson Algorithm for System of Nonlinear Equations

$$x^2 + xy = 10$$
$$y + 3xy^2 = 57$$

$$f_1(x, y) = x^2 + xy - 10 = 0$$
$$f_2(x, y) = y + 3xy^2 - 57 = 0$$

- Jacobian Matrix

$$J = \begin{bmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} \\ \frac{\partial f_2}{\partial x} & \frac{\partial f_2}{\partial y} \end{bmatrix} = \begin{bmatrix} 2x + y & x \\ 3y^2 & 1 + 6xy \end{bmatrix}$$

- Initial guess (let's use the same guess as before): $x_1 = 2.5;\ \ y_1 = 2.5;$ $f_1(x_1, y_1) = 2.5;\quad f_2(x_1, y_1) = -7.625$
- New Jacobian matrix

$$J(x, y) = \begin{bmatrix} 2(2.5) + 2.5 & 2.5 \\ 3(2.5)^2 & 1 + 6(2.5)(2.5) \end{bmatrix} = \begin{bmatrix} 7.5 & 2.5 \\ 18.75 & 38.5 \end{bmatrix}$$

Revisit Example from Slide 12: Newton-Raphson Solution

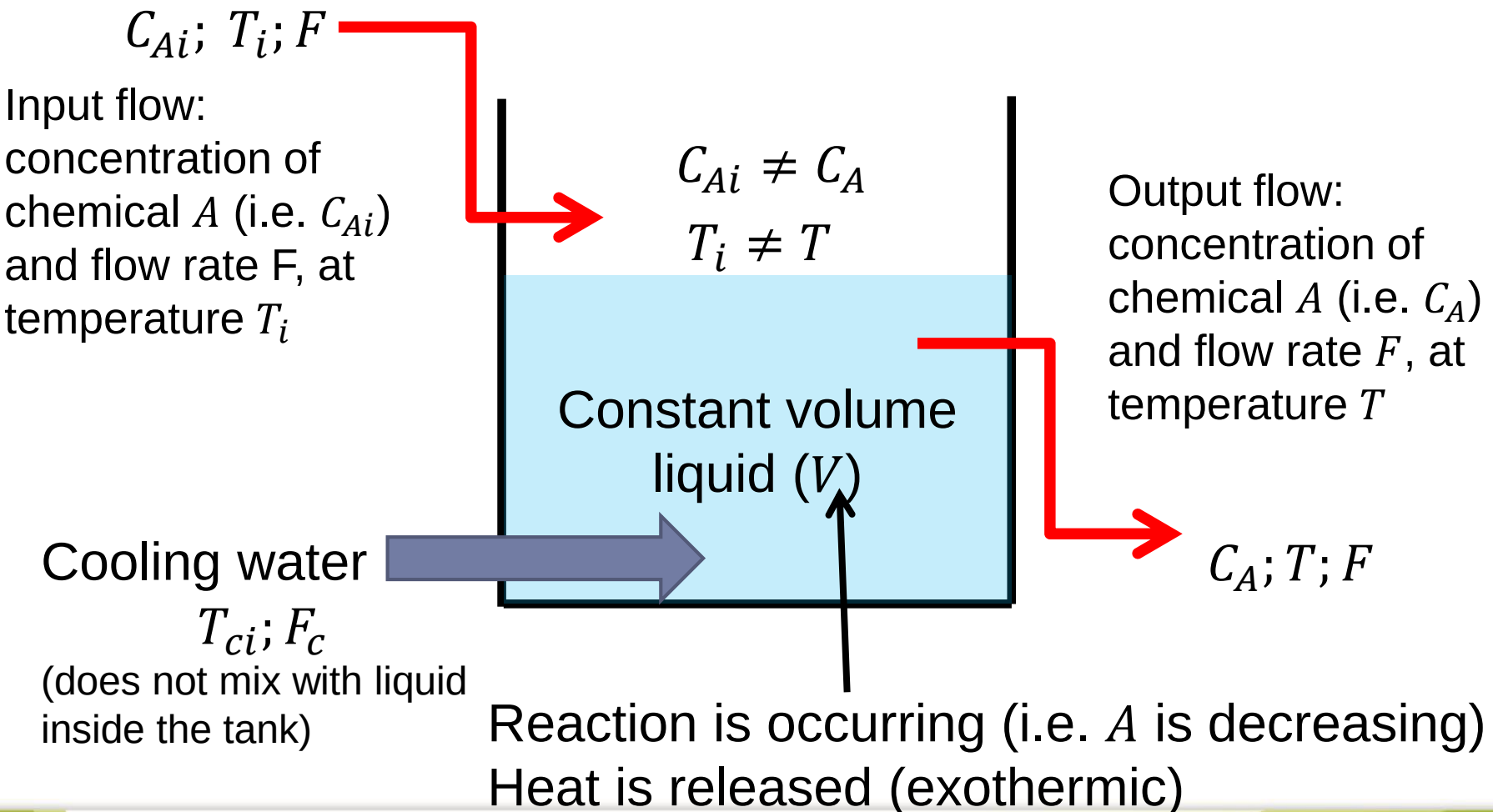
- 1st iteration:

$$\begin{bmatrix} 7.5 & 2.5 \\ 18.75 & 38.5 \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \end{bmatrix} = \begin{bmatrix} -2.5 \\ 7.625 \end{bmatrix} \quad \rightarrow \begin{bmatrix} s_1 \\ s_2 \end{bmatrix} = \begin{bmatrix} -0.4767 \\ 0.4302 \end{bmatrix}$$

$$\begin{bmatrix} x_2 \\ y_2 \end{bmatrix} = \begin{bmatrix} s_1 \\ s_2 \end{bmatrix} + \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} = \begin{bmatrix} 2.0233 \\ 2.9302 \end{bmatrix}$$

Iteration number	1 (initial guess)	2	3	4
y	2.5	2.0233	1.9998	2.0000
x	2.5	2.9302	3.0002	3.0000

Example 2: Tank Reactor: What is the Temperature and the Reactant Concentration in the Tank?



Tank Reactor: Materials Balance

Assume steady-state

$$0 = \{rate\ of\ component\ "A"\ in\} - \{rate\ of\ component\ "A"\ out\} + \{rate\ of\ component\ "A"\ produced\}$$

$$0 = FC_{Ai} - FC_A - Vr$$

$$r = kC_A = k_o e^{-\frac{E}{RT}} C_A$$

$$0 = FC_{Ai} - FC_A - Vk_o e^{-\frac{E}{RT}} C_A$$

1st order reaction (exponential decay)

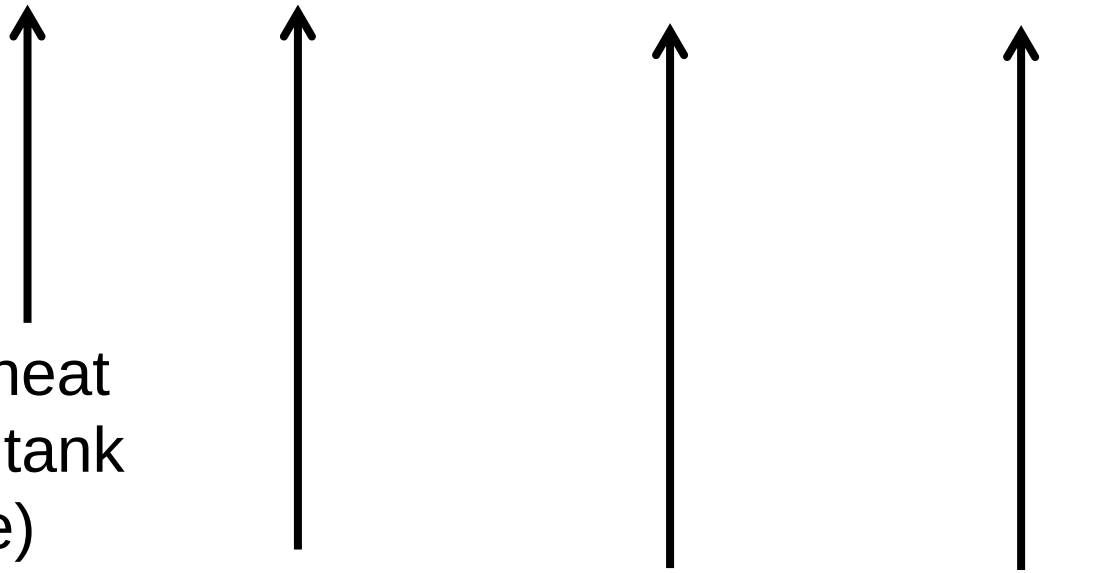
Unknowns are C_A and T.

We need one more equation to solve for C_A and T

Tank Reactor: Energy Balance

Assume steady-state

$$0 = F\rho C_p T_i - F\rho C_p T - A(T - T_{ci}) - \Delta H_R V k_r C_A$$



Flux of heat into the tank (positive)

Flux of heat coming out the tank (negative)

Cooling (negative)

Heat of reaction (positive, $\Delta H_R < 0$)

Tank Reactor: Materials + Energy Balance

We plug in the data

$$F = \frac{1m^3}{min};\ \mathrm{V} = 1m^3;\ C_{Ai} = \frac{2.0kmole}{m^3};\ T_i = 323K;\ C_p = 1cal/(g\ K)$$

$$\rho = \frac{10^6 g}{m^3};\ k_o = 10^{10} min^{-1}; \frac{E}{R} = 8330.1K;\ \Delta H_R = -1.3 \times \frac{10^8 cal}{kmole};$$

$$T_{ci} = 365K; A = 5.3417 \times 10^6 cal/(K\ min)$$

Materials Balance

$$0 = 2 - C_A - 10^{10} e^{\frac{-8330.1}{T}} C_A$$

Energy Balance

$$0 = 3.23 \times 10^8 - 10^6 T - 5.3417 \times 10^6 (T - 365) + 1.3 \times 10^{18} e^{\frac{-8330.1}{T}} C_A$$

$$0 = 323 - T - 5.3417(T - 365) + 1.3 \times 10^{12} e^{\frac{-8330.1}{T}} C_A$$

Newton-Raphson Algorithm for System of Nonlinear Equations

$$f_1(T, C_A) = 2 - C_A - 10^{10} e^{-\frac{8330.1}{T}} C_A = 0$$

$$f_2(T, C_A) = 323 - T - 5.3417(T - 365) + 1.3 \times 10^{12} e^{-\frac{8330.1}{T}} C_A$$

- Jacobian Matrix

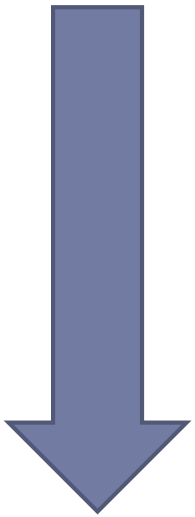
$$J = \begin{bmatrix} \frac{\partial f_1}{\partial T} & \frac{\partial f_1}{\partial C_A} \\ \frac{\partial f_2}{\partial T} & \frac{\partial f_2}{\partial C_A} \end{bmatrix} = \begin{bmatrix} -8.3301 \times 10^{13} e^{-\frac{8330.1}{T}} \frac{C_A}{T^2} & -1 - 10^{10} e^{-\frac{8330.1}{T}} \\ -1 - 5.3417 + 1.0829 \times 10^{16} e^{-\frac{8330.1}{T}} \frac{C_A}{T^2} & 1.3 \times 10^{12} e^{-\frac{8330.1}{T}} \end{bmatrix}$$

- Initial Guess

Let us choose, $C_A = C_{Ai} = \frac{2kmole}{m^3} and\ T = T_{cin} = 365K$

Newton-Raphson Algorithm Results

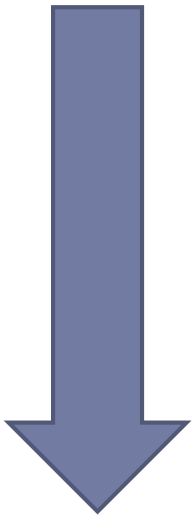
Iteration Number	T (Kelvin)	C_A (kmole/m^3)	f_1	f_2
1 (initial guess)	365	2	-2.4518	276.7289
2	326.2495	3.5673	-1.8581	241.5433
3	388.1774	0.5463	-1.1623	151.1046
4	396.6136	0.1347	0.8468	-110.0878
5	394.2850	0.2483	0.0934	-12.1432
6	393.9580	0.2643	0.0016	-0.2109
7	393.9521	0.2646	5.17e-07	-6.8e-05
8	393.9521	0.2646	5.53e-14	-3.1e-10



Converging Behavior!!!

Newton-Raphson Algorithm Results: Not All is Well Depending on the Initial Guess

Iteration Number	T (Kelvin)	C_A (kmole/m^3)	f_1	f_2
1 (initial guess)	323 (= Ti)	2	-0.1261	240.7427
2	369.9165	1.4371	-1.8230	236.9948
3	252.0485	7.1872	-5.1875	674.3458
4	358.4701	1.9955	-1.6096	209.2500
5	260.2790	6.7856	-4.7864	622.2206
6	358.5943	1.9894	-1.6117	209.5158
7	261.0729	6.7469	-4.7478	617.1964
8	358.6119	1.9885	-1.6120	209.5536



Oscillating Behavior!!!

Not converging!!

Exothermic Tank Reactor: Plotting Temperature

- This system of nonlinear equations is simple enough such that we can eliminate one variable (i.e. C_A) analytically.
- From materials balance equation..

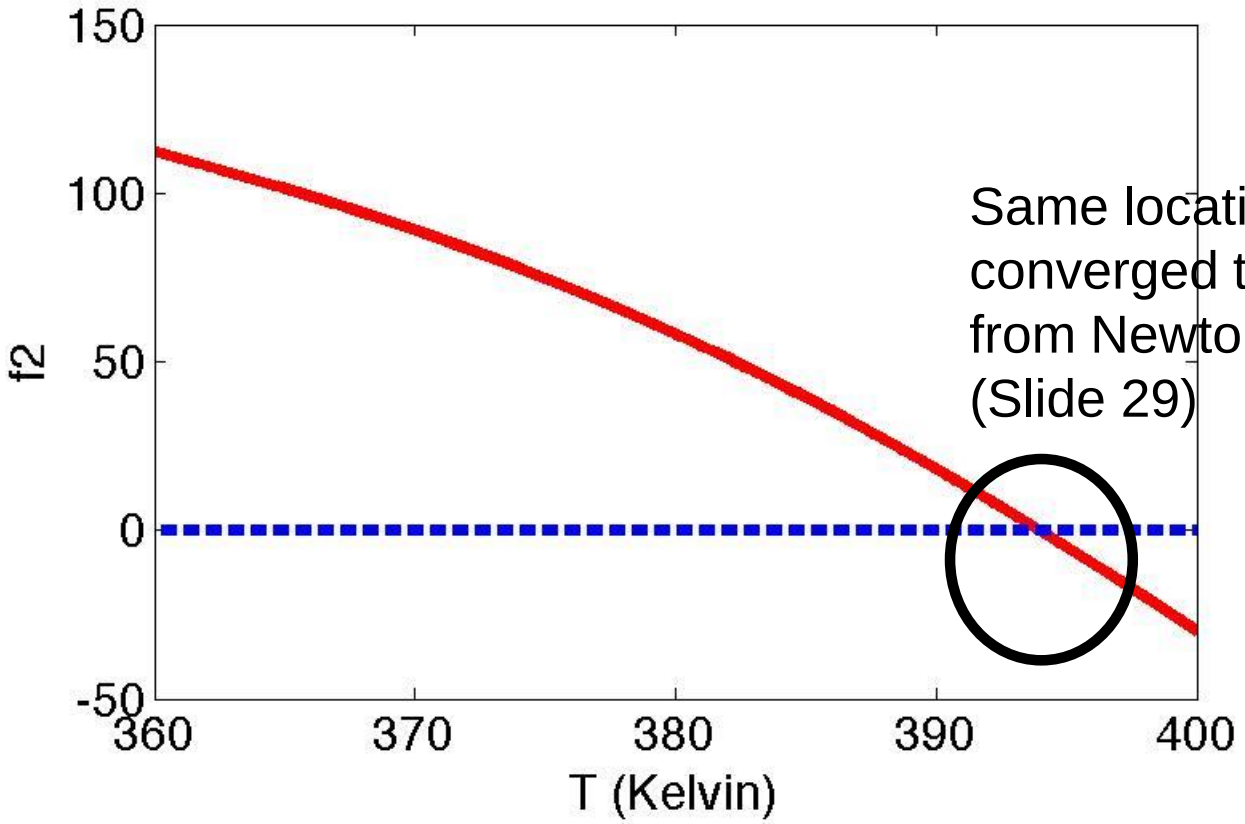
$$f_1(T, C_A) = 2 - C_A - 10^{10} e^{-\frac{8330.1}{T}} C_A = 0$$

$$C_A = \frac{2}{(1 + 10^{10} e^{-\frac{8330.1}{T}})}$$

- Plug C_A into the energy balance equation and plot

$$f_2(T, C_A) = 323 - T - 5.3417(T - 365) + 1.3 \times 10^{12} e^{-\frac{8330.1}{T}} C_A$$

Tank Reactor: Plotting Temperature



- You can obtain C_A from the T value above.
- In general, solving system of nonlinear equations is not so easy such that you can isolate for individual variables. Need to use numerical methods.

Tank Reactor: Changes in the Input Parameters

We plug in the new data

$$F = \frac{1m^3}{min};\ \mathrm{V} = 1m^3;\ C_{Ai} = \frac{2.0kmole}{m^3};\ T_i = 343K;\ C_p = 1cal/(g\ K)$$

$$\rho = \frac{10^6 g}{m^3};\ k_o = 10^{10} min^{-1};\frac{E}{R} = 8330.1K;\ \Delta H_R = -1.3 \times \frac{10^8 cal}{kmole};$$

$$T_{ci} = 310K; A = 1.8736 \times 10^6 cal/(K\ min)$$

$$f_1(T, C_A) = 2 - C_A - 10^{10} e^{-\frac{8330.1}{T}} C_A = 0 \quad (same)$$

$$f_2(T, C_A) = 343 - T - 1.8736(T - 310) + 1.3 \times 10^{12} e^{-\frac{8330.1}{T}} C_A$$

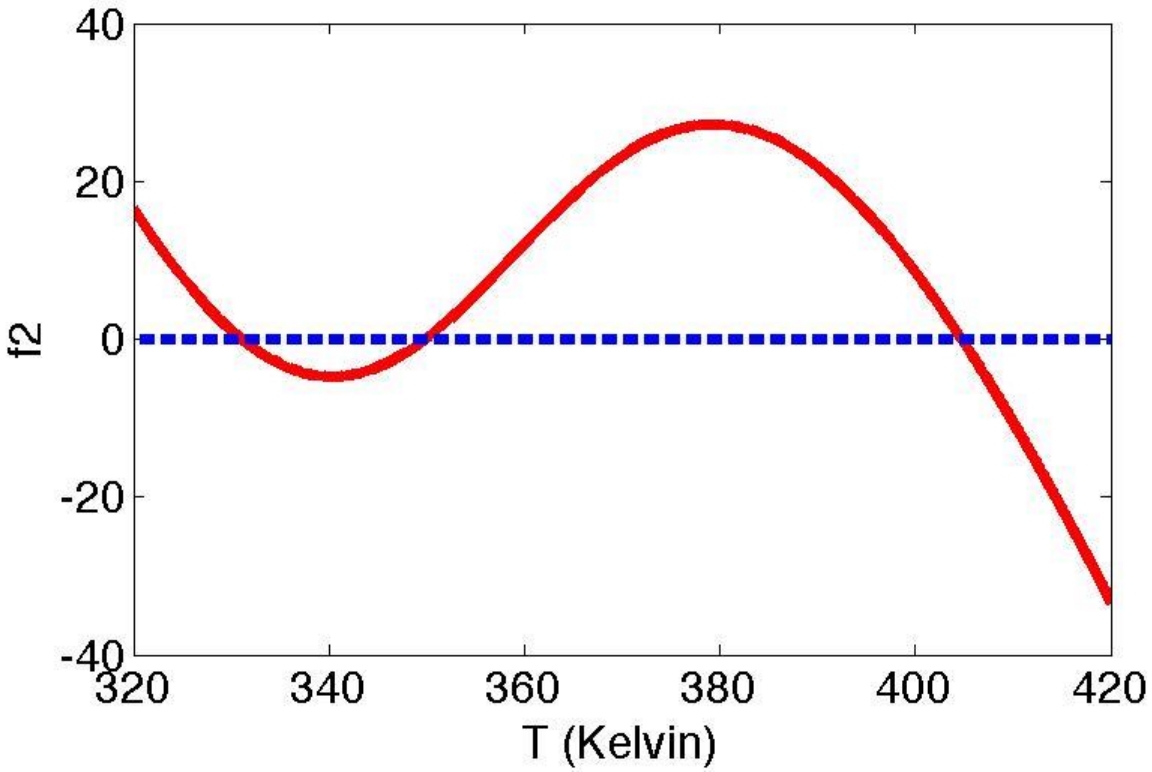
You can try to compute the Jacobian matrix yourself.

Tank Reactor: Changes in the Input Parameters Yield Multiple Solutions

Iteration Number	T (Kelvin)	C_A (kmole/m^3)	T (Kelvin)	C_A (kmole/m^3)	T (Kelvin)	C_A (kmole/m^3)
1	300	2	350	2	400	2
2	324.0064	1.9442	349.9662	1.3704	407.8040	0.2845
3	329.8970	1.8140	349.9032	1.3718	405.1967	0.0919
4	330.9634	1.7905	349.9029	1.3718	404.7500	0.1496
5	331.0092	1.7894	349.9029	1.3718	404.7380	0.1594
6	331.0093	1.7894	349.9029	1.3718	404.7380	0.1597

3 sets of solutions.. All valid mathematically.

Tank Reactor: Plotting Temperature for New Input



- In general, system of nonlinear equations have multiple solutions.
- Newton-Raphson method in this case can find all three set of solutions.

Summary

- System of nonlinear equations are very difficult to solve by hand.
- We can use the open methods (preferably Newton-Raphson over fixed-point iteration) to numerically solve the system of nonlinear equations.
- We should always check to see whether the Newton-Raphson algorithm leads to correct solutions (in many cases, it does not).
- We can try different starting guesses to see the convergent behaviors.
- In engineering, it is always important to make sure if the values we obtain makes sense physically (not just mathematically).

Homework #3 Due 10/17 Thursday Before 10:30 AM

- In this assignment you will code your version of system of nonlinear equations.
- You will turn in three m files:
  - CBE206_hw3_studentID.m

- In the file, you will write the solution to example 1 (Lecture 8, Slide 12). This function will take two input variables x1, and y1.

Homework #3 Due 10/17 Thursday Before 10:30 AM

- For the example, we will make usage of symbolic differentiation in MATLAB.
- Example:

```
clear all;
syms x y    % declare x and y to be variables in symbols to use
f(x,y) = x + cos(y)  % now you can use x and y as variables
df_dx(x,y) = diff(f,x)   % you can differentiate f(x,y) with respect to variable x
df_dy(x,y) = diff(f,y)   % you can differentiate f(x,y) with respect to variable y

a = double(df_dx(1,1))  % you can plug in specific x and y values to df_dx to get the value.  Keyword “double” is used to go from the symbol to actual numerical values that we can use

b = double(df_dy(1,-2))  % similar to above
```

Homework #3 Due 10/17 Thursday Before 10:30 AM

- There are two sets of solutions for this problem: (x_{1_sol}, y_{1_sol}) and (x_{2_sol}, y_{2_sol})
- You will construct an outer loop that finds all these two solutions with initial random guesses for x -> [-5, 5] and y -> [-5, 5]
- The Newton-Raphson algorithm loop will continue until the norm is less than 1e-5. Where, norm is defined as follows: $n = \sqrt{(\overline{s_i})^2}$
- If the Newton-Raphson algorithm does not converge after 100 iterations, we will stop and terminate the loop.
- For the system of linear equations part, you should use the algorithm we learned for the Gaussian elimination section.
- You shouldn't print out duplicate solutions.. So only two sets of solutions (one for first solution and one for second solution) should be printed out (see next page).

Homework #3 Due 10/17 Thursday Before 10:30 AM

- Sample output: (the real solutions are not included below)

```
>> CBE206_hw3_studentID
my initial guesses are 0.688237 and -0.306094
my 1th interation norm =  150.958974
my 2th interation norm =  51.475145
my 3th interation norm =  33.528067
my 4th interation norm =  22.606942
my 5th interation norm =  15.158048
my 6th interation norm =  10.106712
my 7th interation norm =  6.688932
my 8th interation norm =  4.358628
my 9th interation norm =  2.728125
my 10th interation norm =  1.525711
my 11th interation norm =  0.622189
my 12th interation norm =  0.113876
my 13th interation norm =  0.003725
my 14th interation norm =  0.000004
my solutions are x1_sol and y1_sol

my initial guesses are -4.880979 and -1.628774
my 1th interation norm =  2.140872
my 2th interation norm =  19.846354
my 3th interation norm =  26.060261
my 4th interation norm =  8.544744
my 5th interation norm =  3.918265
my 6th interation norm =  1.543379
my 7th interation norm =  0.464519
my 8th interation norm =  0.077482
my 9th interation norm =  0.001295
my 10th interation norm =  0.000000
my solutions are x2_sol and y2_sol
```

Homework #3 Due 10/17 Thursday Before 10:30 AM

- You will be graded on following aspects of the code.

- No error in executing your file
- Obtain both solutions correctly
- Print out the norm values for each iteration
- Print out only two unique solutions (do not print more than two)