Math 6610

- Today's topic: Newton's Method for systems of nonlinear equations
- Let's start with a single equation:

\[ f(x) = 0 \]

\[ f(x) \approx f(x_0) + f'(x_0)(x-x_0) \]

**Local linearization, key idea**

\[ f(x_0) + f'(x_0)(x-x_0) = 0 \]

\[ \Rightarrow x = x_0 - \frac{f(x_0)}{f'(x_0)} \]

- Fixed point iteration

\[ x_0 \text{ given} \quad x_{k+1} = g(x_k) = x_k - \frac{f(x_k)}{f'(x_k)} \quad k = 0, 1, 2, \ldots \]
- Now consider a system
  \[ F(X) = 0 \quad F(X) = \begin{bmatrix} F_1(x) \\ F_2(x) \\ \vdots \\ F_n(x) \end{bmatrix} x \in \mathbb{R}^n \]
  \[ F : \mathbb{R}^n \to \mathbb{R}^n \]

- We apply the same idea
  \[ \nabla F(X) = \begin{bmatrix} \frac{\partial F_1}{\partial x_1}(x) \\ \frac{\partial F_1}{\partial x_2}(x) \\ \vdots \\ \frac{\partial F_n}{\partial x_1}(x) \\ \vdots \\ \frac{\partial F_n}{\partial x_n}(x) \end{bmatrix} \]
  \[ i, j = 1, \ldots, n \]

- We use subscripts to denote components of vectors and superscripts for exponents.

- Need something else for iterates, superscripts in square brackets
  \[ x^{[i]} \text{ given} \]
  \[ \sqrt{\text{Jacobian (matrix)}} \]

  \[ F(x) \approx F(x^{[i]}) + \nabla F(x^{[i]}) (x - x^{[i]}) = L(x) \]

  \[ L(x) = 0 \Rightarrow x = x^{[i]} - \left( \nabla F(x^{[i]}) \right)^{-1} F(x^{[i]}) \]

  \[ x^{[k+1]} = x^{[k]} - \left( \nabla F(x^{[k]}) \right)^{-1} F(x^{[k]}) \]

  \[ k = 0, 1, 2, \ldots \]
As a practical matter (more on this later) we don't invert a matrix.

Instead we solve a linear system,

\[ x^{[0]} \text{ given (depends on the problem)} \]

For \( k = 0, 1, 2, \ldots \), until satisfied,

\[ J^{[k]} = \nabla F(x^{[k]}) \]

\[ \text{solve} \quad J^{[k]} s^{[k]} = -F(x^{[k]}) \]

\[ x^{[k+1]} = x^{[k]} + s^{[k]} \]

\( s^{[k]} \) is the "Newton Step"

(*) is Newton's Method for nonlinear systems.

Generally speaking, any iterative method based on local linearization might be called "Newton's Method" so how does this play out for our GPS problem?
Suppose we have data from 4 satellites, \( S_0, S_1, S_2, S_3 \).

Our unknowns are:

- \( x_v \) location of vehicle
- \( t_v \) time vehicle receives

We can eliminate \( t_v \) and get

three equations:

\[
\begin{align*}
\| x_{S_0} - x_v \| - \| x_{S_i} - x_v \| &= c(t_{S_i} - t_{S_0}) \\
& i = 1, 2, 3
\end{align*}
\]

- \( x_{S_i} \) location of satellite at broadcast
- \( t_{S_i} \) time that satellite broadcast

Starting point:

At first step: location of \( S_2 \)

Every subsequent step: location at last step

There is no general rule of how to get an initial approximation for Newton's Method but for any particular problem, there is usually an obvious choice.
- When do we stop?
- We want accuracy of 1cm.
- Good choice: stop when $\| S^{(k)} \|_1 \leq 10^{-2}$
- This is right at the limits of floating point accuracy, you may have to relax this requirement in your receiver.

- But: we probably have data from more than four satellites.
- We don't want to throw away that info!

- General Discussion, you want to translate this into the specifics of the term project.
- Suppose we want to solve
  \[ F(x) = 0 \quad F: \mathbb{R}^n \rightarrow \mathbb{R}^m \quad m > n \]
- We have \( m \) equations in \( n \) unknowns, \( m > n \)
- The system is overdetermined and we usually won't have a solution.
- Instead solve
  \[ f(x) = F(x)^T F(x) = \min \]
- \( f(x) \geq 0 \) and if \( F(x) = 0 \) \( f(x) = 0 \)
- So instead of making \( f(x) = 0 \) which will usually be impossible, we make it as small as possible.
- This is Discrete Nonlinear Least Squares
- How do we solve (**)?
- Solve the nonlinear system
  \[ \nabla f(x) = 0 \]
- Gradient
- How do we do that?
- why, we use Newton's Method!
- So we need to compute the Jacobian of \( \nabla f \)

\[
J(x) = \nabla^2 f = \begin{bmatrix}
\frac{\partial^2 f}{\partial x_i \partial x_j}
\end{bmatrix}
\]

- \( J(x) \) is symmetric and (usually) positive definite.

- In our particular least squares problem (\( n = 3 \))

\[
f(x) = F(x)^T F(x) \quad \text{\( F: m \times 1 \)}
\]

\[
\nabla f(x) = 2 \begin{bmatrix}
\nabla F(x)
\end{bmatrix}^T F(x)
\]

\[
\nabla F(x) : \ n \times m
\]

\[
\nabla f(x) : \ n \times 1
\]

\[
J(x) = 2 \begin{bmatrix}

\nabla^2 F(x)^T F(x) + (\nabla F(x))^T \nabla F(x)
\end{bmatrix}
\]
\( \nabla^2 F(x) \) is an \( n \times n \times m \times m \) tensor of second order partial derivatives of the components of \( F \)

- But: \( \| F(x) \| \) is going to be small. It's safe to ignore this term.

- So run a variation of Newton's Method using

\[
J(x) = 2 (\nabla F(x))^T \nabla F(x)
\]

- \( J(x) \) is symmetric.

\[
A = 2 B^T B
\]

\[
x^T A x = 2 x^T B^T B x = 2 y^T y \quad y = B x
\]

and positive definite (unless \( B \) is rank-deficient).

- General subject: Fixed Point Iteration.