Math 128A, Summer 2019
Lecture 9, Monday 7/8/2019

1 Discussion: Homework 2

1.1 Question 4

For 4(b), recall the homework says our interval (α, β) will depend on a.

Aside: Newton's method is affinely invariant (that is, scalar multiplication to $f(x)$ cancels out with its derivative). To see this further, see P. Deuflhard for an in-depth discussion on Newton's method.

For 4(c), consider:

$$g(x) = x(2 - ax)$$
$$|g'(x)| = |2 - 2ax| \leq \frac{1}{2}$$

We're interested in the above line for convergence. So we have:

$$-\frac{1}{2} \leq 2 - 2ax \leq \frac{1}{2}$$
$$\frac{5}{4a} \geq x \geq \frac{3}{4a}$$

So the rate of convergence is quadratic, **because**

$$g'\left(\frac{1}{a}\right) = 0.$$

Aside: In general math, we just care about radius of convergence; however, for applied math, we need fast convergence. Hence we look for a better bound:

$$|g'(x)| \leq \frac{1}{2}$$

1.2 Review: Rate of Convergence

Consider $x_n \to x$. Then we can write

$$|x_n - x| \leq K\theta^n = O(\theta^n), \qquad \theta < 1$$

This gives linear convergence with ratio θ. Sometimes linear convergence is good (and we'd take it where it's **guaranteed**). For example, in bisection, if we start with an interval with the root bracketed.
If our initial error $|x_0 - x| < 1$, then we have quadratic convergence:

$$|x_n - x| \leq K|x_0 - x|^{2^n}.$$

That is,

$$\frac{|x_{n+1} - x|}{|x_n - x|^2} \leq K.$$

Remark: Remember that there's a difference between **cost** and **precision** for big-O notation. We say $W(\delta)$ "work" for cost and δ for precision.

So in our example, $\delta := \theta^n$, we have $n|\log\theta| = |\log\delta|$. Then we'd have:

$$\begin{aligned} W(\delta) = n &= O(|\log\delta|) \\ n &= \log\delta/\log\theta \\ &= 52 \end{aligned}$$

Out in the real world, how do we get K and θ?
We estimate θ by:

$$\begin{aligned} \frac{K\theta^{n+1}}{K\theta^n}\frac{|x_{n+1}-x|}{|x_n-x|} &= \frac{|x_{n+1}-x_n+x_n-x|}{|x_n-x_{n-1}+x_{n-1}-x|} \\ \theta &\approx \frac{|x_{n+1}-x_n|+|x_n-x|}{|x_n-x_{n-1}|+|x_{n-1}-x|} \\ &= \frac{K\theta^n+\delta_{n+1}}{K\theta^{n-1}+\delta_n} \\ &= \theta\left(\frac{K\theta^{n-1}+\delta_{n+1}/\theta}{K\theta^{n-1}+\delta_n}\right) \end{aligned}$$

So conclude:

$$\begin{aligned} 1 &= \frac{K\theta^{n-1}+\delta_n+(\delta_{n+1}/\theta-\delta_n)}{K\theta^{n-1}+\delta_n} \\ \implies 0 &= \delta_{n+1}/\theta-\delta_n \\ \theta &= \frac{\delta_{n+1}}{\delta_n} \end{aligned}$$

Remark: Strain gives an idea that for `fsolve( ..., x0)`, we can feed our found `x0` as the initial guess for `fsolve`.
For question 6, coding Newton's, we'll want a **diverse selection** of starting points that lead to different behavior.

2 Polynomial Interpolation

Consider a polynomial

$$P(t) = a_0 + a_1 t + \cdots + a_n t^n.$$

Why polynomials? They're easy to work with, and the parameters occur linearly, and they have a massive algebraic structure. However, for numerical analysis, we usually want to represent then in bases that are NOT the standard monomials.
In our first definition with the standard monomial basis, our polynomial lives (is centered) at 0, so we call it "zero-centric". So we can build another polynomial as follows, for $t_0 \neq t_1 \neq \cdots \neq t_n$:

$$P(t) = b_0 + b_1(t-t_0) + b_2(t-t_0)(t-t_1) + \cdots + b_n(t-t_0)(t-t_1)\cdots(t-tn-1)$$

We stop at $n-1$ index on the last term to ensure our polynomial is degree n. Consider that all of our terms in the second definition vanish at $t := t_0$,

besides b_0. And likewise, the others for other t_i. So this gives us some "triangular" representation of the polynomial.
The general goal is to approximate an arbitrary function $f(t)$ by some polynomial $P(t)$, and then we perform some operation that we want(ed) to do to f, but we do that instead to $P(t)$. For example, differentiating or integrating; if we don't know how to do this for f, we do it on the $P(t)$.
That is,

$$f'(t) \approx p'(t); \qquad \int_a^b f(t)\ dt \approx \int_a^b p(t)\ dt.$$

And of course, we can do this with other basis functions, such as trigonometric functions (as in Fourier analysis), but it turns out to be more or less the same thing.
Hence, interpolation is the **type** of approximation (which is perhaps the most famous) but not the only one. That is, we build some polynomial that matches $P(x) = f(x)$ at some set of points $x_0, \ldots, x_n$. This gives entirely different systems of equations, depending on what basis we choose.

$$\begin{aligned} a_0 + a_1 t_0 + a_2 t_0^2 + \cdots + a_n t_0^n &= f(t_0) \\ &\vdots \\ a_0 + a_1 t_n + a_2 t_n^2 + \cdots + a_n t_n^n &= f(t_n) \end{aligned}$$

This gives a $(n+1) \times (n+1)$ system of linear equations

$$\begin{bmatrix} 1 & t_0 & \cdots & t_0^n \\ \vdots & \ddots & & \vdots \\ 1 & t_n & \cdots & t_n^n \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} f_0 \\ \vdots \\ f_n \end{bmatrix}$$

where our conditions of solvability of this equation depends on this left matrix. So our left matrix is invertible IF AND ONLY IF $t_0 \neq t_1 \neq \cdots \neq t_n$, as known by the determinant.
An alternative to proving our matrix is invertible (determinant nonzero) is to show that the nullspace is nontrivial. Or alternatively, we can use:

$$1, (t - t_0), (t - t_0)(t - t_1), \ldots$$

This is known as the Newton basis, which gives us the triangular pattern as we saw before, where terms vanish for certain inputs. To see this explicitly, consider:

$$\begin{aligned} p(t_0) &= b_0 = f_0 \\ p(t_1) &= b_0 + b_1(t_1 - t_0) = f_1 \\ p(t_2) &= b_0 + b_1(t_2 - t_0)(t_2 - t_0)(t_2 - t_1) = f_2 \\ &\vdots \end{aligned}$$

So this implies there is a polynomial $p(t)$ interpolating $f(t)$ at $n+1$ points $t_0, \ldots, t_n$. This gives a lower-diagonal matrix, where if our main diagonal entries are nonzero, the matrix is invertible.

Recall from linear algebra, transformations for change-of-basis. A particularly useful result is:

$$\det B = \det(X)\det(A)\frac{1}{\det(X)},$$

where the determinant is **invariant** under the change-of-basis transformation. To conclude, we express the same requirements, but just in a different basis (now Newton basis).
To put this concretely, for example:

```
n = 0, constant
  p(t) = f(t_0)
n = 1 linear
  p(t0) = f(t0)
  p(t1) = f(t1)
```

Consider:

$$\begin{bmatrix}1 & t_0\\ 1 & t_1\end{bmatrix}\begin{bmatrix}a_0\\ a_1\end{bmatrix} = \begin{bmatrix}f_0\\ f_1\end{bmatrix}$$

Let $p(t) = b_0 + b_1(t - t_0)$. We have:

$$\begin{bmatrix}1 & 0\\ 1 & (t_1 - t_0)\end{bmatrix}\begin{bmatrix}b_0\\ b_1\end{bmatrix} = \begin{bmatrix}f_0\\ f_1\end{bmatrix}$$

This gives $b_0 = f_0$ and

$$\begin{aligned} b_0 + (t_1 - t_0)b_1 &= f_1\\ (t_1 - t_0)b_1 &= f_1 - f_0\\ b_1 &= \frac{f_1 - f_0}{t_1 - t_0}\end{aligned}$$

Hence we conclude:

$$\begin{aligned} p(t) = b_0 + b_1(t - t_0) &= f_0 + \frac{f_1 - f_0}{t_1 - t_0}(t - t_0)\\ &= \frac{f_0(t_1 - t_0) + (f_1 - f_0)(t - t_0)}{t_1 - t_0}\\ &= f_0\frac{t_1 - t}{t_1 - t_0} + f_1\frac{t_0 - t}{t_0 - t},\end{aligned}$$

which is known as the Lagrange form (Lagrangian Interpolation). In the above, we define:

$$\text{divided difference} := \frac{f_1 - f_0}{t_1 - t_0}.$$

Lagrange discovered the (fairly obvious) generalization for a "Lagrange ba-

sis" of degree n:

$$\begin{aligned}
L_0(t) &:= \frac{(t-t_1)(t-t_2)\cdots(t-t_n)}{(t_0-t_1)(t_0-t_2)\cdots(t_0-t_n)} \\
L_1(t) &:= \frac{(t-t_0)(t-t_2)\cdots(t-t_n)}{(t_1-t_0)(t_1-t_2)\cdots(t_1-t_n)} \\
&\vdots \\
L_j(t) &= \frac{\Pi_{k\neq j}(t-t_k)}{\Pi_{k\neq j}(t_j-t_k)} \\
&\vdots \\
L_n(t) &:= \frac{(t-t_1)(t-t_2)\cdots(t-t_{n-1})}{(t_n-t_1)(t_n-t_2)\cdots(t_n-t_{n-1})}
\end{aligned}$$

So we conclude, in the Lagrange basis,

$$\begin{aligned}
p(t) &= f_0L_0(t) + f_1L_1(t) + \cdots + f_nL_n(t) \\
&= \sum_{j=0}^{n} \left[f_jL_j(t)\right], \qquad L_j := \frac{\Pi_{k\neq j}(t-t_k)}{\Pi_{k\neq j}(t_j-t_k)}.
\end{aligned}$$

If we define $\omega_j := \frac{1}{\Pi_{k\neq j}(t_j-t_k)}$, we can write:

$$\begin{aligned}
p(t) &= \sum_{j=0}^{n} f_j\omega_j \frac{\Pi_k(t-t_k)}{t-t_j} \\
&= \omega(t)\sum_{j=0}^{n} \frac{f_j\omega_j}{t-t_j}.
\end{aligned}$$

This is called the **Barycentric form**, and it has its uses.
To continue our example earlier, we can write:

$$f_0\left(\frac{t_1-t}{t_1-t_0}\right) + f_1\left(\frac{t_0-t}{t_0-t_1}\right) = \underbrace{(t_0-t)(t_1-t)}_{\omega(t)}\left[\frac{f_0}{t_1-t_0}\frac{1}{t_0-t} + \frac{f_1}{t_0-t_1}\frac{1}{t_1-t}\right].$$

3 Error in Polynomial Interpolation

For example, if we interpolate with $n = 0$ (constant function f) then we have:

$$f(t) - p(t) = f(t) - f(t_0) = f'(\xi)(t-t_0),$$

by the mean value theorem. This error estimate (formula) is exactly the error for n-degree interpolation. Consider that $(t-t_0)$ makes the error equal to 0 at $t = t_0$. Also, $f'(\xi)$ is all about f, so if f is a constant, then the derivative is zero, and thus the error is identically zero across.
Then for linear interpolation $n = 1$:

$$f(t) - p(t) = \underbrace{[c\cdot f''(\xi)]}_{0 \text{ when } f \text{ is linear}} \cdot(t-t_0)(t-t_1)$$

Or equivalently, having two distinct points gives a unique line. To find the constant c, we test it on the function on the right, $\omega(t) := (t - t_0)(t - t_1)$. Hence

$$\begin{aligned} f(t) &= \omega(t) \\ f(t_0) &= f(t_0) = 0 \\ \implies p(t) &= 0, \end{aligned}$$

the linear polynomial going through points at zero, the zero function. Then equivalently write

$$\begin{aligned} \omega(t) &= t^2 + \cdots + \cdots + 1 \\ \omega''(t) &= 2! \end{aligned}$$

And we conclude $c := \frac{1}{2!}$ for above. Hence the error for **linear** interpolation is precisely:

$$f(t) - p(t) = \frac{1}{2!} f''(\xi) \left[(t - t_0)(t - t_1) \right]$$

So we want to sample close to the evaluation point to minimize the right side bracketed expression.

Lecture ends here.

Next time we'll see what goes wrong with interpolation.