Ch 8.4: Approximation with Rational Functions

Monday, October 27, 2025 10:22 AM

Rather than approximate a function f (or interpolate data) with a <u>polynomial</u> $p(x)$, let's use a rational function $r(x) = \frac{p(x)}{q(x)}$. ← polynomial ← polynomial

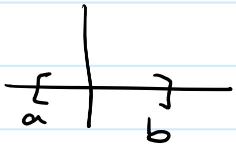
let $p(x)$ have degree n
$q(x)$ have degree m and $N = n+m$

<u>Why?</u>

Possibly less oscillation than polynomials, maybe more accuracy w/ fewer terms N, especially if f has a singularity just outside our interval of interest.

<u>Setup</u>: $$r(x) = \frac{p(x)}{q(x)} = \frac{p_0 + p_1 x + \dots + p_n x^n}{q_0 + q_1 x + \dots + q_m x^m}$$

If we're interested in approximating f over $[a,b]$,



and wlog assume $[a,b]$ is shifted so $a < 0 < b$, we don't want $r(0) = \infty$ so require $q_0 \neq 0$.

In fact, $$r(x) = \frac{p_0/q_0 + p_1/q_0\, x + \dots + p_n/q_0\, x^n}{q_0/q_0 + q_1/q_0\, x + \dots + q_m/q_0\, x^m}$$ i.e. take $q_0 = 1$ WLOG.

Padé Approximation

Idea: use the Taylor series approx. of f (about 0) } so a <u>local</u> approximation, <u>not</u> uniform on an interval

Say, $f(x) = \sum_{i=0}^{\infty} a_i x^i$, so $a_i = \frac{1}{i!} f^{(i)}(0)$

Write

$$\underbrace{f(x) - r(x)}_{\text{error in our approximation... so want this small}} = f(x) - \frac{p(x)}{r(x)} = \frac{f(x)\,r(x) - p(x)}{q(x)}$$

$$= \frac{\left(\sum_{i=0}^{\infty} a_i x^i\right)\left(\sum_{j=0}^{m} q_j x^j\right) - \left(\sum_{k=0}^{n} p_k x^k\right)}{q(x)}$$ } ignore denominator

We can't guarantee $f(x) - r(x) = 0$ for all x, but we could force $f(0) - r(0) = 0$.
Even better, also $f'(0) - r'(0) = 0$, etc.

forcing $f(0)=r(0)$, $f'(0)=r'(0)$, $\dots$, $f^{(N)}(0)=r^{(N)}(0)$ is equivalent to forcing that numerator

$$(a_0+a_1x+\dots)(1+q_1x+\dots+q_mx^m)-(p_0+p_1x+\dots+p_nx^n)$$

to have no terms of degree $\leq N$.

$$\underbrace{a_0\cdot 1} + \underbrace{a_0q_1x + a_1\cdot 1\cdot x} + \dots \underbrace{-p_0}\ \underbrace{-p_1x} - p_2x^2$$

cancel

cancel

Defining $p_k=0$ if $k>n$
$q_k=0$ if $k>m$, we must solve

$$\sum_{i=0}^{k} a_i q_{k-i} = p_k \quad \text{for } k=0,1,\dots,N \Big\}$$ $N+1$ equations.

$\{q_i, p_k\}$ $N+1$ unknowns (since $q_0=1$)

Ex: $f(x)=e^{-x}$, $n=1$, $m=1$ (see book for $n=3, m=2$)

$m=0$ is boring since then $q(x)=1$, $p(x)$ is Taylor

numerator is $(1-x+x^2/2-x^3/6)(1+q_1x)-(p_0+p_1x)$

so:

$$1-p_0=0 \Rightarrow \boxed{p_0=1}$$

$$1\cdot q_1x - x - p_1x = 0 \Rightarrow p_1=q_1-1$$

$$-q_1x^2+x^2/2 = 0 \to \boxed{q_1=1/2} \to \text{so } p_1=-1/2$$

So

$$r(x)=\frac{1-\frac12 x}{1+\frac12 x}$$

is our Padé approx. of e^{-x} of order $(1,1)$ about $x=0$

Note: you can write $r(x)$ as a continued fraction for faster evaluation

Beyond Padé

① Chebyshev: similar computation to Padé (cancel out terms in numerator) but use $f(x) = \sum_{k=0}^{\infty} a_k T_k(x)$ ← kth Chebyshev polynomial

instead of its Taylor series.

Also write $r(x) = \dfrac{p(x)}{q(x)} = \dfrac{\sum_{k=0}^{n} p_k T_k(x)}{\sum_{k=0}^{m} q_k T_k(x)}$ in Chebyshev basis

Can combine with the Remez algorithm to get good minimax approximation

② AAA = adaptive Antoulas-Anderson (Y. Nakatsuka, O. Sète, L.N. Trefethen, 2018)

Now in Scipy!

Doesn't write $r(x) = \dfrac{p(x)}{q(x)}$ but instead as the barycentric quotient

$$r(x) = \frac{n(x)}{d(x)} = \frac{\sum_{j=1}^{m} \frac{w_j f(s_j)}{x-s_j}}{\sum_{j=1}^{m} \frac{w_j}{x-s_j}}$$

having sampled f at the "support points" /nodes $\{s_j\}_{j=1}^{m}$

You give it a domain and it chooses points $\{s_j\}$ and computes the weights

Works very well!