Polynomial Approximation: Pros and Cons.

Advantages of Polynomial Approximation:

1. We can approximate any continuous function on a closed interval to within arbitrary tolerance. (*Weierstrass approximation theorem*)
2. Easily evaluated at arbitrary values. (*e.g.* *Horner’s method*)
3. Derivatives and integrals are easily determined.

Disadvantage of Polynomial Approximation:

1. Polynomials tend to be oscillatory, which causes errors. This is sometimes, but not always, fixable: — *E.g.* if we are free to select the node points we can minimize the interpolation error (*Chebyshev polynomials*), or optimize for integration (*Gaussian Quadrature*).
Padé Approximation

Extension of Taylor expansion to rational functions; selecting the $p_i$'s and $q_i$'s so that $r^{(k)}(x_0) = f^{(k)}(x_0)$ $\forall k = 0, 1, \ldots, N$.

$$f(x) - r(x) = f(x) - \frac{p(x)}{q(x)} = \frac{f(x)q(x) - p(x)}{q(x)}$$

Now, use the Taylor expansion $f(x) \sim \sum_{i=0}^{\infty} a_i (x - x_0)^i$, for simplicity $x_0 = 0$:

$$f(x) - r(x) = \frac{\sum_{i=0}^{\infty} a_i x^i \sum_{i=0}^{m} q_i x^i - \sum_{i=0}^{n} p_i x^i}{q(x)}.$$  

Next, we choose $p_0, p_1, \ldots, p_n$ and $q_1, q_2, \ldots, q_m$ so that the numerator has no terms of degree $\leq N$.

Note: $p_0 = a_0$!! (This reduces the number of unknowns and equations by one (1).)

Padé Approximation: The Mechanics.

For simplicity we (sometimes) define the “indexing-out-of-bounds” coefficients:

$$\begin{cases} p_{n+1} = p_{n+2} = \cdots = p_N = 0 \\ q_{m+1} = q_{m+2} = \cdots = q_N = 0, \end{cases}$$

so we can express the coefficients of $x^k$ in

$$\sum_{i=0}^{\infty} a_i x^i \sum_{i=0}^{m} q_i x^i - \sum_{i=0}^{n} p_i x^i = 0, \quad k = 0, 1, \ldots, N$$

as

$$\sum_{i=0}^{k} a_i q_{k-i} = p_k, \quad k = 0, 1, \ldots, N.$$
Padé Approximation: Concrete Example, $e^{-x}$

The Taylor series expansion for $e^{-x}$ about $x_0 = 0$ is $\sum_{k=0}^{\infty} \frac{(-1)^k}{k!} x^k$, hence $\{a_0, a_1, a_2, a_3, a_4, a_5\} = \{1, -1, \frac{1}{2}, -\frac{1}{6}, \frac{1}{24}, -\frac{1}{120}\}$.

\[
\begin{bmatrix}
1 & 0 & -1 \\
-1 & 1 & 0 & -1 \\
1/2 & -1 & 0 & 0 & -1 \\
-1/6 & 1/2 & 0 & 0 & 0 \\
1/24 & -1/6 & 0 & 0 & 0 \\
\end{bmatrix}
\begin{bmatrix}
q_1 \\
q_2 \\
p_1 \\
p_2 \\
p_3 \\
\end{bmatrix} =
\begin{bmatrix}
-1 \\
1/2 \\
-1/6 \\
1/24 \\
-1/120 \\
\end{bmatrix},
\]

which gives $\{q_1, q_2, p_1, p_2, p_3\} = \{2/5, 1/20, -3/5, 3/20, -1/60\}$, i.e.

\[r_{3,2}(x) = \frac{1 - \frac{3}{5}x + \frac{3}{20}x^2 - \frac{1}{60}x^3}{1 + \frac{2}{5}x + \frac{1}{20}x^2}.
\]

Note: $r_{5,0}(x)$ is the Taylor polynomial of degree 5.

The algorithm in the book looks frightening! If we think in term of the matrix problem defined earlier, it is easier to figure out what is going on:

```matlab
% The Taylor Coefficients, a_0, a_1, a_2, a_3, a_4, a_5
a = [1 1 1/2 -1/6 1/24 -1/120]';
N = length(a); A = zeros(N-1,N-1);
% m is the degree of q(x), and n the degree of p(x)
m = 3; n = N-1-m;
% Set up the columns which multiply q_1 through q_m for i=1:m
A(i:(N-1),i) = a(1:(N-i));
end
% Set up the columns that multiply p_1 through p_n
A(1:n,m+1:(1:n)) = -eye(n)
% Set up the right-hand-side
b = - a(2:N);
% Solve
c = A \ b;
Q = [1; c(1:m)]; % Select q_0 through q_m
P = [a_0; c((m+1):m+n)]; % Select p_0 through p_n
```
Optimal Padé Approximation?

<table>
<thead>
<tr>
<th>Polynomials</th>
<th>One Point</th>
<th>Optimal Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>Taylor</td>
<td>Polynomial Basis Functions</td>
<td></td>
</tr>
<tr>
<td>Chebyshev</td>
<td>???</td>
<td></td>
</tr>
<tr>
<td>Padé</td>
<td>Rational Basis Functions</td>
<td></td>
</tr>
</tbody>
</table>

From the example $e^{-x}$ we can see that Padé approximations suffer from the same problem as Taylor polynomials — they are very accurate near one point, but away from that point the approximation degrades.

“Chebyshev-placement” of interpolating points for polynomials gave us an optimal (uniform) error bound over the interval.

Can we do something similar for rational approximations???

Chebyshev Basis for the Padé Approximation!

We use the same idea — instead of expanding in terms of the basis functions $x^k$, we will use the Chebyshev polynomials, $T_k(x)$, as our basis, i.e.

$$r_{n,m}(x) = \frac{\sum_{k=0}^{n} p_k T_k(x)}{\sum_{k=0}^{m} q_k T_k(x)}$$

where $N = n + m$, and $q_0 = 1$.

We also need to expand $f(x)$ in a series of Chebyshev polynomials:

$$f(x) = \sum_{k=0}^{\infty} a_k T_k(x),$$

so that

$$f(x) - r_{n,m}(x) = \frac{\sum_{k=0}^{\infty} a_k T_k(x) \sum_{k=0}^{m} q_k T_k(x) - \sum_{k=0}^{n} p_k T_k(x)}{\sum_{k=0}^{m} q_k T_k(x)}.$$

Example: Revisiting $e^{-x}$ with Chebyshev-Padé Approximation

The 8th order Chebyshev-expansion (all praise Maple) for $e^{-x}$ is

$$P_8^{CT}(x) = 1.266065878 T_0(x) - 1.130318208 T_1(x) + 0.2714953396 T_2(x)$$

and using the same strategy — building a matrix and right-hand-side utilizing the coefficients in this expansion, we can solve for the Chebyshev-Padé polynomials of degree $(n + 2m) \leq 8$:

Next slide shows the matrix set-up for the $r_{3,2}^{CT}(x)$ approximation.

Note: Due to the “folding”, $T_i(x) T_j(x) = \frac{1}{2} \left[ T_{i+j}(x) + T_{|i-j|}(x) \right]$, we need $n + 2m$ Chebyshev-expansion coefficients. (Burden-Faires do not mention this, but it is “obvious” from algorithm 8.2; Example 2 (p. 519) is broken, – it needs $P_7(x)$.)
Example: Revisiting \( e^{-x} \) with Chebyshev-Padé Approximation

\[
T_0(x) : \frac{1}{2} \left[ a_1 q_1 + a_2 q_2 - 2p_0 = 2a_0 \right]
\]

\[
T_1(x) : \frac{1}{2} \left[ (2a_0 + a_2)q_1 + (a_1 + a_3)q_2 - 2p_1 = 2a_1 \right]
\]

\[
T_2(x) : \frac{1}{2} \left[ (a_1 + a_3)q_1 + (2a_0 + a_4)q_2 - 2p_2 = 2a_2 \right]
\]

\[
T_3(x) : \frac{1}{2} \left[ (a_2 + a_4)q_1 + (a_1 + a_5)q_2 - 2p_3 = 2a_3 \right]
\]

\[
T_4(x) : \frac{1}{2} \left[ (a_3 + a_5)q_1 + (a_2 + a_6)q_2 - 0 = 2a_4 \right]
\]

\[
T_5(x) : \frac{1}{2} \left[ (a_4 + a_6)q_1 + (a_3 + a_7)q_2 - 0 = 2a_5 \right]
\]

Example: Revisiting \( e^{-x} \) with Chebyshev-Padé Approximation

\[
R_{CP}^{CP}(x) = \frac{1.155054 T_0(x) - 0.8549674 T_1(x) + 0.1561297 T_2(x) - 0.01713502 T_3(x) + 0.001066402 T_4(x)}{T_0(x) + 0.1964246628 T_1(x)}
\]

\[
R_{CP1}^{CP}(x) = \frac{1.05053166 T_0(x) - 0.6016362122 T_1(x) + 0.07418797149 T_2(x) - 0.00410958353 T_3(x)}{T_0(x) + 0.387050965 T_1(x) + 0.02365167312 T_2(x)}
\]

\[
R_{CP2}^{CP}(x) = \frac{0.9541897238 T_0(x) - 0.373756255 T_1(x) + 0.02331049609 T_2(x)}{T_0(x) + 0.5682932066 T_1(x) + 0.06011746318 T_2(x) + 0.003726440404 T_3(x)}
\]

\[
R_{CP3}^{CP}(x) = \frac{0.8671327116 T_0(x) - 0.1731320271 T_1(x)}{T_0(x) + 0.73743710 T_1(x) + 0.13373746 T_2(x) + 0.014470654 T_3(x) + 0.00086486509 T_4(x)}
\]
The Chebyshev basis does not give an optimal (in the min-max sense) rational approximation. However, the result can be used as a starting point for the second Remez algorithm. It is an iterative scheme which converges to the best approximation.

A discussion of how and why (and why not) you may want to use the second Remez’ algorithm can be found in Numerical Recipes in C: The Art of Scientific Computing (Section 5.13). [You can read it for free on the web\(^*\) — just Google for it!]

\(^*\) The old 2nd Edition is Free, the new 3rd edition is for sale...