Numerical Analysis Project 1

Margaret Dorsey

September 17, 2016

Positive Solutions of X

Claim: There is exactly one positive solution for x.
Proof:

We first note $f(0) < 0$ for all values of k ,η, ξ - when $x = 0$, the rest of the function $f(x)$ disappears, leaving us with $f(0) = -\xi$, a negative value, because ξ represents ligand concentration, a physical quantity, and thus must be positive (because 0 is a trivial case and negative values are non-physical).

Additionally, for $x = \xi$, $f(x) > 0$, because $f(x) = \xi - \xi + \sum_{j=1}^{M} \frac{k_j \eta_j}{1+k_j \eta}$ in this case, where all of the sum terms are non-negative, and at least one is non-zero (otherwise it is modelling a trivial case where there is no amount of any binding molecules).

Thus, by the intermediate value theorem, we know that f has at least one positive root between 0 and ξ. Calculating the derivative with respect to x of f, we get

$$f'(x) = 1 + \sum_{j=1}^{M} = \frac{k_j \eta_j}{(1 + k_j x)^2}$$

which by reasoning analogous to the above argument, is always positive for positive x. Thus $f(x)$ is strictly increasing for positive x, and due to Rolle's theorem, we know that $f(x)$ restricted to positive x has at most one root, which completes the proof.

■

Fixed Point Iteration

Finding $g(x)$

Case Testing

Choosing α

Claim: If $\alpha = \sum_{j=1}^{M} k_j n_j$, the FPI always converges.
Proof:

■

Newton's Method

Test One Output

```
Enter the intial guess: 1.6

--------------------------
 Newton
 -----------------------



i: 0 x: -1.124105012 value: 88.452818065
i: 1 x: -1.260131012 value: 46.182036757
i: 2 x: -1.570535828 value: 24.956847208
i: 3 x: -2.357298970 value: 14.010274589
i: 4 x: -4.536830098 value: 7.290560454
i: 5 x: -8.588461093 value: 1.729329152
i: 6 x: -10.061914568 value: 0.041604994
i: 7 x: -10.099003088 value: 0.000018410
i: 8 x: -10.099019514 value: 0.000000000
```

Test Two Output

Output

```
Enter the intial guess: 1.4999

--------------------------
 Newton
 -----------------------



i: 0 x: -0.999876924 value: -81242.749956933
i: 1 x: -0.999753861 value: -40619.374953876
i: 2 x: -0.999507770 value: -20307.687427792
i: 3 x: -0.999015733 value: -10151.843615770
i: 4 x: -0.998032241 value: -5073.921612311
i: 5 x: -0.996067582 value: -2534.960417718
i: 6 x: -0.992147546 value: -1265.479442812
i: 7 x: -0.984344518 value: -630.738232506
i: 8 x: -0.968885874 value: -313.366310246
i: 9 x: -0.938552168 value: -154.678222468
i: 10 x: -0.880170250 value: -75.331900493
i: 11 x: -0.772155001 value: -35.661641481
```

```
i: 12 x: -0.587979613 value: -15.858623262
i: 13 x: -0.323256320 value: -6.099899665
i: 14 x: -0.056125978 value: -1.650760190
i: 15 x: 0.078909646 value: -0.189707093
i: 16 x: 0.098689910 value: -0.003059278
i: 17 x: 0.099019425 value: -0.000000818
i: 18 x: 0.099019514 value: -0.000000000
```

Test Three Output

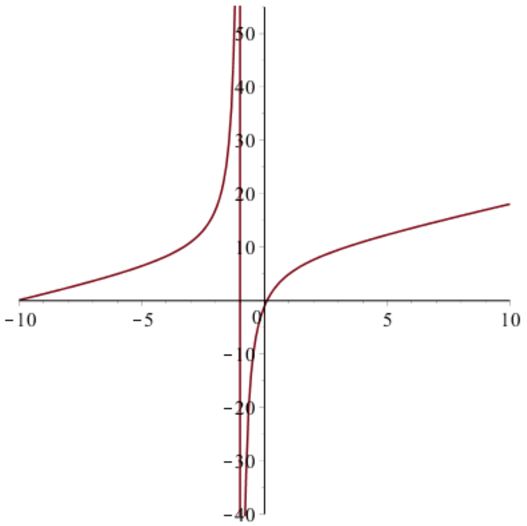
```
Enter the intial guess: 1.5

--------------------------
 Newton
 -----------------------


i: 0 x: -1.000000000 value: -inf
i: 1 x: -nan value: -nan
```

Test Four Output

Analysis



For an initial guess of 1.6, the method successfully converged, however unfortunately it was to the negative root. For 1.499, the method converged (somewhat slowly, for newton's method) to the correct root, however 1.5 leads to division by zero. Examining the graph of f(x), the cause of this behavior becomes fairly apparent - as the positive side of the graph flattens and the slope of the tangent line approaches 0, Newton's method tends to send x_{n+1} through and across the singularity at $x = -1$, into the negative values of x. For x before this jump across the singularity of $f(x)$, such as 1.4999, the method, although displaced into negative x, can self-correct back towards the positive root. As the slope of $f(x)$ gets more and more positive approaching the positive root, Newton's method manages to converge more and more efficiently, as evidenced by the results of initial guess 1.0.

Algorithm Comparison

Output Evaluation

Asymptotic Error Constant Calculation

Test Case Results

Conclusions and Preferred Algorithm