

Thus the conditions for theorem 2.2 (part b) are met, so $g(x)$ has a unique fixed point on $[0, 2\pi]$.

Suppose we start at the interval's midpoint, $p_0 = \pi$. (Note: any $p_0 \in [0, 2\pi]$ is okay.) We can use the error bound $|p_n - p| \leq k^n \cdot \pi$ (π being the maximum distance between p_0 and any point in $[0, 2\pi]$), or compute $p_1 = \pi + 1/2$ and use the error bound $|p_n - p| \leq k^n / (1 - k) \cdot (1/2) = (2/3) \cdot 4^{-n}$. The latter error bound is more favorable, and drops below 10^{-2} when $n = 4$. (The former bound requires $n = 5$.) Actual calculation, shown in the following table, shows that only 3 iterations are needed.

```
>> fixedpoint_table(@(x)pi+0.5*sin(x/2), pi, 1e-3, 10)
```

n	p_n	Delta-p(n-1)
1	+3.64159265	+0.50000000
2	+3.62604886	-0.01554379
3	+3.62699562	+0.00094676

- 11 In each case, we determine an interval $[a, b]$ on which $a \leq g(x) \leq b$ and $|g'(x)| < 1$. We then take $k = \max_{a \leq x \leq b} |g'(x)|$, and conclude that fixed point iteration converges for any $p_0 \in [a, b]$, with error $|p_n - p| \leq k^n \cdot (b - a)$. We finally solve for n making $(b - a)k^n \leq 10^{-5}$ and run `fixedpoint_table` using this n and a tolerance smaller than 10^{-5} .

(One very cautious approach is to use the error bound $|p_n - p| \leq \frac{k}{1-k} |p_1 - p_0|$: this implies that $|p_n - p|$ is accurate to within 10^{-5} as long as $|p_n - p_{n-1}| \leq 10^{-5}(1 - k)/k$. So we may use tolerance $10^{-5}(1 - k)/k$.)

- (d) Since 5^{-x} is decreasing, the first requirement ($a \leq g(x) \leq b$ for $x \in [a, b]$) is the same as saying $5^{-a} \leq b$ and $5^{-b} \leq a$. Since $g'(x) = -\ln(5) \cdot 5^{-x}$, $|g'(x)| < 1$ when $x > \log_5(\ln(5)) \approx 0.2957$. So a must be larger than 0.2957.

If we round up ($a \geq 0.3$), and pick b slightly larger than 5^{-a} (0.7 will do), we find that $5^{-b} \geq a$. So fixed point iteration converges for any $p_0 \in [.3, .7]$, and we may take $k = \ln(5) \cdot 5^{-.3} \approx 0.9931$ in the error bound. This only guarantees convergence after 1531 iterations! However, `fixedpoint_table` should show that the desired accuracy is attained around $n = 40$ (depending on the choice of p_0), and the fixed point is $p \approx 0.46962$.

- (f) We can use the range of this sine wave as our interval. It's convenient to rewrite $g(x)$ as $\sin(x + \pi/4)/\sqrt{2}$. (This is the same, by the angle-sum formula for sin. This step is just a shortcut; we could also have used calculus to find the range of g and g' .) We see that $g(x) \in [-1/\sqrt{2}, 1/\sqrt{2}]$ and $|g'(x)| \leq 1/\sqrt{2}$. So we may choose that $[a, b]$ and k . The error bound drops below 10^{-5} when $n = 33$. If we start with $p_0 = 0$, we find that $p_6 = 0.704812$ is already accurate enough.

2.3

18. The following tables show that bisection gets better results than false position, and the secant method does not even converge to the root. (Setting a tolerance of zero ensures that all ten iterations are run.)

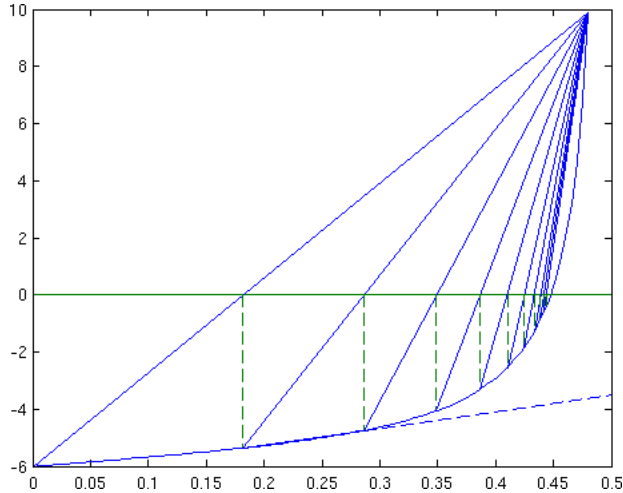
Under good conditions, we would have expected false position to beat bisection, and secant to beat false position. However, this function produces "worst-case" behavior for each algorithm. The reasons for this can be understood with the help of a picture like figure 1.

The method of false position ensures that its estimates bracket a root, but at the cost of leaving the right endpoint ($x = .48$) fixed for the entire calculation! Thus, the method makes secant line approximations to $f(x)$ which are less accurate than the secant line through $(.48, f(.48))$ and $(\arctan(6)/\pi, 0)$, which is already a poor fit. This explains why the method converges so slowly to the root.

In the secant method, the two estimates following 0.48 are fairly small, close together, and (most importantly) on a shallow part of the graph of $f(x)$. Thus, the next secant line crosses the x -axis to the right of the vertical asymptote at $x = .5$, and all hell breaks loose.

```
>> bisection_table(@(x)tan(pi*x)-6,0,.48,0,10)
```

FIGURE 1. Steps taken by the false position and secant methods in 2.3.18: Both methods start by drawing the two leftmost secant lines in the figure. The false position method continues drawing secant lines through $(0.48, 9.9)$ forever. The secant method draws the dashed secant line, following it to a point far to the right of the vertical asymptote in the graph of f .



n	a	b	p_n
1	+0.0000000	+0.4800000	+0.2400000
2	+0.2400000	+0.4800000	+0.3600000
3	+0.3600000	+0.4800000	+0.4200000
4	+0.4200000	+0.4800000	+0.4500000
5	+0.4200000	+0.4500000	+0.4350000
6	+0.4350000	+0.4500000	+0.4425000
7	+0.4425000	+0.4500000	+0.4462500
8	+0.4462500	+0.4500000	+0.4481250
9	+0.4462500	+0.4481250	+0.4471875
10	+0.4471875	+0.4481250	+0.44765625

??? Error using ==> bisection_table at 19
Exceeded maximum number of iterations

```
>> false_position_table(@(x)tan(pi*x)-6,0,.48,0,10)
```

n	x1	x2	p_n
1	+0.0000000	+0.4800000	+0.18119424
2	+0.4800000	+0.18119424	+0.28618717
3	+0.4800000	+0.28618717	+0.34898123
4	+0.4800000	+0.34898123	+0.38705262
5	+0.4800000	+0.38705262	+0.41030472
6	+0.4800000	+0.41030472	+0.42456648
7	+0.4800000	+0.42456648	+0.43333631
8	+0.4800000	+0.43333631	+0.43873741
9	+0.4800000	+0.43873741	+0.44206695
10	+0.4800000	+0.44206695	+0.44412066

??? Error using ==> false_position_table at 28
Exceeded maximum number of iterations

```

>> secant_table(@(x)tan(pi*x)-6,0,.48,0,10)
n |          x1 |          x2 |          p_n
-----
 1 | +0.00000000 | +0.48000000 | +0.18119424
 2 | +0.48000000 | +0.18119424 | +0.28618717
 3 | +0.18119424 | +0.28618717 | +1.09198611
 4 | +0.28618717 | +1.09198611 | -3.69229667
 5 | +1.09198611 | -3.69229667 | -22.60064985
 6 | -3.69229667 | -22.60064985 | -57.22283247
 7 | -22.60064985 | -57.22283247 | +3.53875815
 8 | -57.22283247 | +3.53875815 | -113.94440505
 9 | +3.53875815 | -113.94440505 | -195.89499482
10 | -113.94440505 | -195.89499482 | -2989.94003753
??? Error using ==> secant_table at 25
Exceeded maximum number of iterations

```

2.4

10. We may factor $f(x)$ as $(x-p)^m q(x)$, for some continuous function $q(x)$ with $q(p) \neq 0$. Then

$$g(x) = x - \frac{m(x-p)^m q(x)}{m(x-p)^{m-1} q(x) + (x-p)^m q'(x)} = x - \frac{m(x-p)q(x)}{mq(x) + (x-p)q'(x)}, \text{ and}$$

$$g'(x) = 1 - \frac{m[q(x) + (x-p)q'(x)][mq(x) + (x-p)q'(x)] - [m(x-p)q(x)][mq'(x) + q'(x) + (x-p)q''(x)]}{mq(x) + (x-p)q'(x)}.$$

When $x = p$, this mess simplifies to

$$g'(p) = 1 - \frac{m[q(p)][mq(p)]}{mq(p)},$$

which we may evaluate (as $1 - 1 = 0$) because the $q(p)$ in the denominator is not zero.

14. Assume the sequence $\{p_n\}$ is convergent to p of order α , and that $|p_{n+1} - p| \approx C|p_n - p||p_{n-1} - p|$ for sufficiently large n . Since $|p_{n+1} - p| \approx \lambda|p_n - p|^\alpha$ for large n , we have $\lambda|p_n - p|^\alpha \approx C|p_n - p||p_{n-1} - p|$. But we also have $|p_n - p| \approx \lambda|p_{n-1} - p|^\alpha$, so $\lambda[\lambda|p_{n-1} - p|^\alpha]^\alpha \approx C\lambda|p_{n-1} - p|^\alpha|p_{n-1} - p|$. Thus, $\lambda^\alpha|p_{n-1} - p|^{\alpha^2 - \alpha - 1} \approx C$ for large n . Since $p_{n-1} - p$ converges to zero, this is only possible if $\alpha^2 - \alpha - 1 = 0$. Hence, $\alpha = (1 \pm \sqrt{5})/2$, and indeed α is $(1 + \sqrt{5})/2$ since it cannot be negative.

2.5

17. (a) Note $\frac{p_{n+1} - p}{p_n - p} = \frac{P_{n+1}(x) - e^x}{P_n(x) - e^x} = \frac{-e^{\xi_{n+1}} x^{n+2} / (n+2)!}{-e^{\xi_n} x^{n+1} / (n+1)!} = \frac{e^{\xi_{n+1} - \xi_n} x}{n+2}$, for numbers ξ_k between 0 and x . This goes to zero, slowly as $n \rightarrow \infty$. (Convergence isn't exactly linear; the theorem still works.)

(b) >> % Generate p_n; we need p0 through p10.

```

>> p(1)=1;
>> for n=1:10
p(n+1) = p(n) + 1/factorial(n);
end
>> % Calculate differences.
>> DeltaP = p(2:11) - p(1:10);
>> Delta2P = DeltaP(2:10) - DeltaP(1:9);
>> % Calculate pHat.
>> pHat = p(1:9) - DeltaP(1:9).^2 ./ Delta2P;

```

```

p : 1.0000 2.0000 2.5000 2.6667 2.7083 2.7167 2.7181 2.7183 2.7183 2.7183 2.7183
p^: 3.0000 2.7500 2.7222 2.7187 2.7183 2.7183 2.7183 2.7183 2.7183 2.7183

```

(c) Yes!