Pluto.jl

Save notebook...

```julia
using QuadGK
```

```julia
begin
    function I_of_p_1(p)
        integrand(y) = sqrt((1 - y) / (p^2 + y))
        val, _ = quadgk(integrand, 0.0, 1.0; rtol=1e-6, atol=1e-6)
        return val
    end

    function solve_p0_1(a, b; tol=1e-6, maxiter=50)
        fa = I_of_p_1(a) - 1
        fb = I_of_p_1(b) - 1
        if fa * fb > 0
            error
        end
        for iter in 1:maxiter
            m = (a + b) / 2
            fm = I_of_p_1(m) - 1
            @info "iter=$iter  p=$m   I(p)-1=$(fm)"
            if abs(fm) < tol
                return m
            end
            if fa * fm < 0
                b = m
                fb = fm
            else
                a = m
                fa = fm
            end
        end
        return (a + b) / 2
    end

# -------------------
# Run solver
# -------------------

    p0 = solve_p0_1(-0.5, 0.0)
    println("\nSolution p(0) = $p0")
end
```

```
iter=1  p=-0.25   I(p)-1=0.1586812931495858
iter=2  p=-0.375   I(p)-1=0.00746678908620968
iter=3  p=-0.4375   I(p)-1=-0.057391792381185414
iter=4  p=-0.40625   I(p)-1=-0.02577927908922506
iter=5  p=-0.390625   I(p)-1=-0.009366090740190347
iter=6  p=-0.3828125   I(p)-1=-0.0010028320346118225
iter=7  p=-0.37890625   I(p)-1=0.003218592422551403
iter=8  p=-0.380859375   I(p)-1=0.0011045450552582547
iter=9  p=-0.3818359375   I(p)-1=5.002414580479453e-5
iter=10  p=-0.38232421875   I(p)-1=-0.00047661185821501473
iter=11  p=-0.382080078125   I(p)-1=-0.0002133458568137625
iter=12  p=-0.3819580078125   I(p)-1=-8.167385842694497e-5
iter=13  p=-0.38189697265625   I(p)-1=-1.5828107387982904e-5
iter=14  p=-0.381866455078125   I(p)-1=1.7097206395977338e-5
iter=15  p=-0.3818817138671875   I(p)-1=6.343463063718247e-7

Solution p(0) = -0.3818817138671875
```

430 ms

```julia
begin
    using Printf

    # Computes the interval value at midpoint
    function I_of_p(p)
        integrand(y) = sqrt((1 - y) / (p^2 + y))
        val, err = quadgk(integrand, 0.0, 1.0; rtol=1e-6, atol=1e-6)
        return val, err
    end

    function solve_p0(a, b; tol=1e-10, maxiter=50)
        val_a, _ = I_of_p(a)
        fa = val_a - 1
        val_b, _ = I_of_p(b)
        fb = val_b - 1

        if fa * fb > 0
            error("Bisection interval does not bracket the root.")
        end

        for iter in 1:maxiter
            m = (a + b) / 2
            val, err = I_of_p(m)
            fm_interval_lo = val - err
            fm_interval_hi = val + err

            @printf("Iter %d: m = %.10f   Integral[m] ∈ [%.12f, %.12f]\n", iter, m,
fm_interval_lo, fm_interval_hi)

            fm = val - 1
            if abs(fm) < tol
                println("Converged at iter $iter: root ≈ $m; Integral ∈
[$fm_interval_lo, $fm_interval_hi]")
                return a, b, (fm_interval_lo, fm_interval_hi)
            end

            if fa * fm < 0
                b = m
                fb = fm
            else
                a = m
                fa = fm
            end
        end

        println("Max iterations reached. Interval [$a, $b], last Integral ∈
[$fm_interval_lo, $fm_interval_hi]")
        return a, b, (fm_interval_lo, fm_interval_hi)
    end

    # Run
    p_a, p_b, (lo, hi) = solve_p0(-0.5, 0.0)
    println("\nRoot lies in [$p_a, $p_b] with last Integral ∈ [$lo, $hi]")
end
```

```
Iter 1: m = -0.2500000000   Integral[m] ∈ [1.158680663755, 1.158681922544]
Iter 2: m = -0.3750000000   Integral[m] ∈ [1.007466235589, 1.007467342584]
Iter 3: m = -0.4375000000   Integral[m] ∈ [0.942607776390, 0.942608638847]
Iter 4: m = -0.4062500000   Integral[m] ∈ [0.974220256956, 0.974221184865]
Iter 5: m = -0.3906250000   Integral[m] ∈ [0.990633412092, 0.990634406428]
Iter 6: m = -0.3828125000   Integral[m] ∈ [0.998996646396, 0.998997689535]
Iter 7: m = -0.3789062500   Integral[m] ∈ [1.003218055968, 1.003219128877]
Iter 8: m = -0.3808593750   Integral[m] ∈ [1.001104016294, 1.001105073816]
Iter 9: m = -0.3818359375   Integral[m] ∈ [1.000049499041, 1.000050549251]
Iter 10: m = -0.3823242188   Integral[m] ∈ [0.999522864820, 0.999523911464]
Iter 11: m = -0.3820800781   Integral[m] ∈ [0.999786129933, 0.999787178353]
Iter 12: m = -0.3819580078   Integral[m] ∈ [0.999917801485, 0.999918850798]
Iter 13: m = -0.3818969727   Integral[m] ∈ [0.999983647012, 0.999984696773]
Iter 14: m = -0.3818664551   Integral[m] ∈ [1.000016572214, 1.000017622199]
Iter 15: m = -0.3818817139   Integral[m] ∈ [1.000000109410, 1.000001159283]
Iter 16: m = -0.3818893433   Integral[m] ∈ [0.999991878160, 0.999992927977]
Iter 17: m = -0.3818855286   Integral[m] ∈ [0.999995993772, 0.999997043617]
Iter 18: m = -0.3818836212   Integral[m] ∈ [0.999998051588, 0.999999101447]
Iter 19: m = -0.3818826675   Integral[m] ∈ [0.999999080498, 1.000000130364]
Iter 20: m = -0.3818821907   Integral[m] ∈ [0.999999594954, 1.000000644823]
Iter 21: m = -0.3818824291   Integral[m] ∈ [0.999999337726, 1.000000387593]
Iter 22: m = -0.3818823099   Integral[m] ∈ [0.999999466340, 1.000000516208]
Iter 23: m = -0.3818822503   Integral[m] ∈ [0.999999530647, 1.000000580516]
Iter 24: m = -0.3818822801   Integral[m] ∈ [0.999999498493, 1.000000548362]
Iter 25: m = -0.3818822950   Integral[m] ∈ [0.999999482417, 1.000000532285]
Iter 26: m = -0.3818823025   Integral[m] ∈ [0.999999474378, 1.000000524247]
Iter 27: m = -0.3818822987   Integral[m] ∈ [0.999999478398, 1.000000528266]
Iter 28: m = -0.3818823006   Integral[m] ∈ [0.999999476388, 1.000000526256]
Iter 29: m = -0.3818823015   Integral[m] ∈ [0.999999475383, 1.000000525251]
Iter 30: m = -0.3818823020   Integral[m] ∈ [0.999999474881, 1.000000524749]
Iter 31: m = -0.3818823018   Integral[m] ∈ [0.999999475132, 1.000000525000]
Converged at iter 31: root ≈ -0.3818823017645627; Integral ∈ [0.999999475131918
8, 1.0000005250002422]

Root lies in [-0.38188230199739337, -0.3818823015317321] with last Integral ∈ [
0.9999994751319188, 1.0000005250002422]
```

```julia
begin

    f(x) = sin(x + exp(x))
    val, err = quadgk(f, 0.0, 8.0)
    @printf("Integral : %.10f\n", val)
    @printf("Error bound: %.10e\n", err)
    @printf("Interval containing the integral: [%.10f, %.10f]\n", val - err, val +
    err)
end
```

```
Integral : 0.3474001727
Error bound: 5.0976381923e-09
Interval containing the integral: [0.3474001676, 0.3474001778]
```

English

How can we make better? Instant feedback... Send

Live docs Status