In [12]:
```
using Interact
using Gadfly
```

In [2]:
```
set_default_plot_size(25cm,10cm)
```

Chapter 3 - Root Finding Algorithms

1. Bisection

Example

The function $bisect$ uses the bisection algorithm to find a root of f in the [a,b] interval up to tol precision. it outputs the value of x such that |x-x0|<tol, where x0 is the solution.

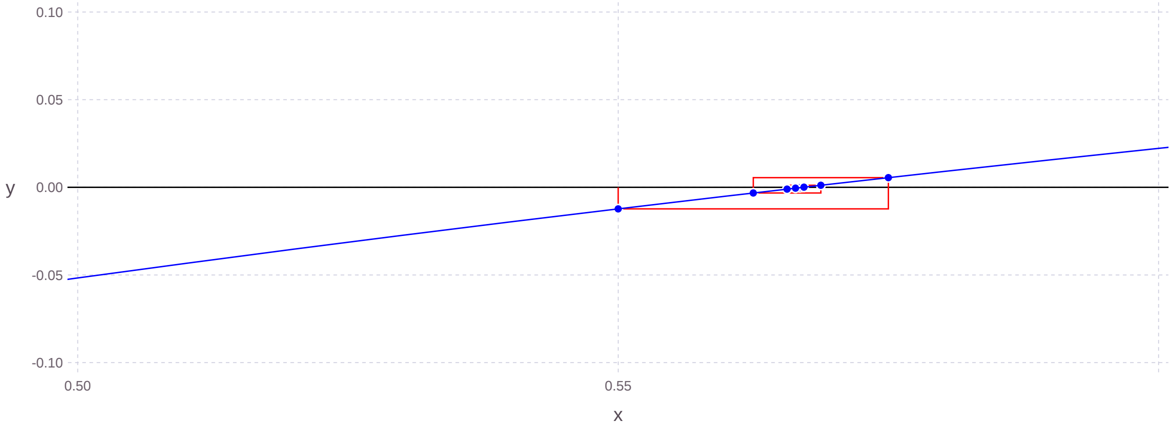
In [3]:
```
function example_Bissection(f, a, b, tol)

    maxiter = 1000
    iter = 0

    results = ["iter" "a" "b" "x" "f(x)" "error"]

    while b-a > tol && iter <= maxiter
        iter+=1
        results = [results; iter a b ((a+b)/2) f((a+b)/2) ((b-a)-tol)]
        if f(a)*f((a+b)/2) > 0 a=(a+b)/2
            else b=(a+b)/2 end
    end

    x = 0.45:0.01:0.65
    roots = results[2:size(results)[1],4]

    iter_points = [results[2, 4] 0; results[2, 4] results[2, 5]]
    for i in 1:(size(results)[1]-2)
        iter_points = [iter_points; results[i+2, 4] results[i+1, 5]; results[
    end

    p = plot(
        layer(x=x, y=f.(x), Geom.line, Theme(default_color=colorant"blue"), c
        layer(x=roots, y=f.(roots), Geom.point, Theme(default_color=colorant"
        layer(x=iter_points[:,1], y=iter_points[:,2], Geom.line(preserve_orde
        layer(x=[0.4 0.7], y=[0 0], Geom.line, Theme(default_color=colorant"b
        Coord.Cartesian(xmin=0.5,xmax=.6)
    )

    return results, [results[1:4,:]; results[(size(results)[1]-2):size(result
end;
```

In [4]:
```
f(p) = 1-0.5*p^(-0.5)-0.3*p^(-0.2)
example_Bissection(f, 0.5, 0.6, 10.0^-15)[2]
```

Out[4]:
```
7×6 Array{Any,2}:
   "iter"   "a"       "b"       "x"        "f(x)"        "error"
  1.0      0.5       0.6       0.55      -0.0123026     0.1
  2.0      0.55      0.6       0.575      0.0055093     0.05
  3.0      0.55      0.575     0.5625    -0.00325321    0.025
 45.0      0.567094  0.567094  0.567094   3.88578e-16   4.66214e-15
 46.0      0.567094  0.567094  0.567094  -6.66134e-16   1.88658e-15
 47.0      0.567094  0.567094  0.567094  -1.66533e-16   4.4329e-16
```

In [5]:
```
example_Bissection(f, 0.5, 0.6, 10.0^-15)[3]
```

Out[5]:



2. Fixed Point Function Iteration

Exercise 6.

Let $f(x) = cos(x)$. Find x such that $f(x) = x$ up to 10^{-10} precision.

In [6]:
```
function ex6_FixedPoint(x0)

    f(x) = cos(x)

    path = [x0 0; x0 f(x0)]
    results = ["iter" "x" "cos(x)"]
    iter = 0

    for i in 1:100
        x0 = f(x0); iter += 1
        results = [results; iter x0 cos(x0)]
        path = [path; x0 x0; x0 f(x0)]
    end

    x = (-pi/2):0.01:(pi/2)
    p = plot(
        layer(x=[x[1] x[length(x)]], y=[0 0], Geom.line, Theme(default_color=
        layer(x=x, y=f.(x), Geom.line, Theme(default_color=colorant"blue"), o
        layer(x=x, y=x, Geom.line, Theme(default_color=colorant"green"), orde
        layer(x=path[:,1], y=path[:,2], Geom.line(preserve_order=true), Theme
        Coord.Cartesian(xmin=x[1],xmax=x[length(x)],ymin=0,ymax=1)
    )

    return results, [results[1:4,:]; results[(size(results)[1]-2):size(result
end;
```

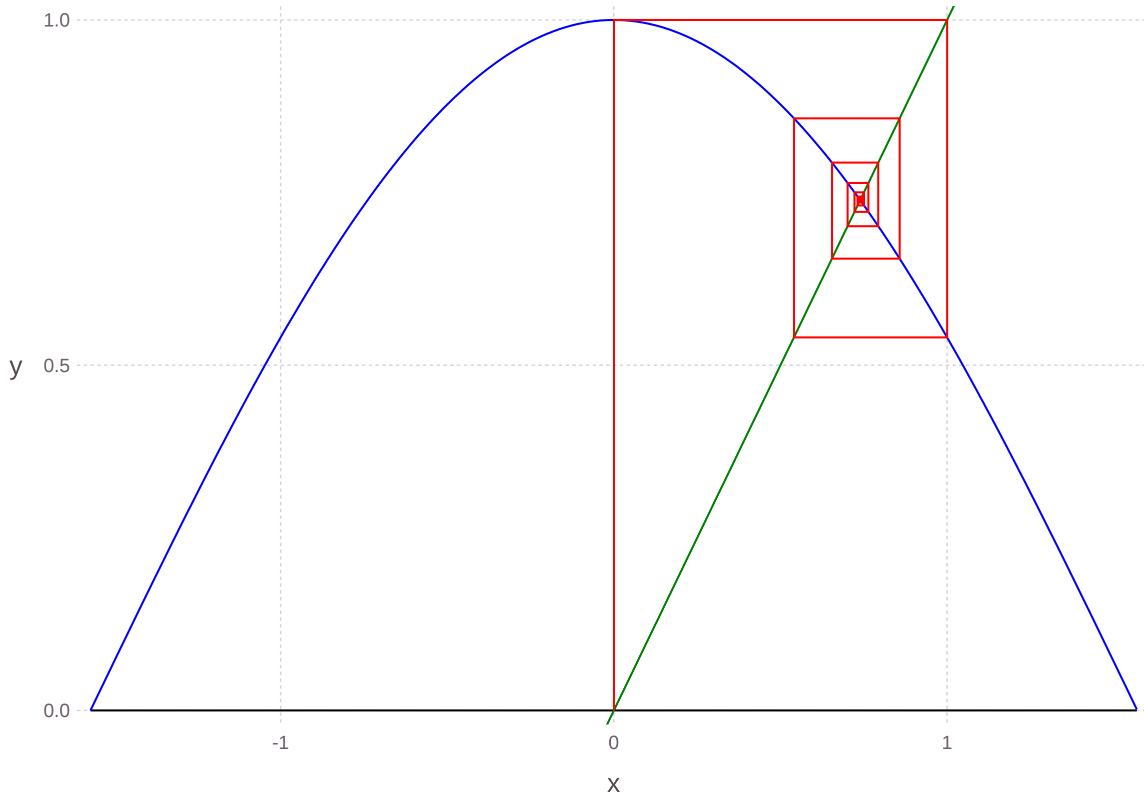
In [7]:
```
ex6_FixedPoint(1)[2]
```

Out[7]:
```
7×3 Array{Any,2}:
    "iter"   "x"       "cos(x)"
   1.0      0.540302  0.857553
   2.0      0.857553  0.65429
   3.0      0.65429   0.79348
  98.0      0.739085  0.739085
  99.0      0.739085  0.739085
 100.0      0.739085  0.739085
```

In [8]:
```
set_default_plot_size(18cm,13cm)

@manipulate for x_0 in (-1.5):0.1:(1.5)
    ex6_FixedPoint(x_0)[3]
end
```

x_0 0.0

Out[8]:



3. Newton-Rhapson

Exercise 7.

Let $f(x) = 1 - 0.5x^{-0.5} - 0.3x^{-0.2}$. Find x: $f(x) = 0$.

In [9]:

```
function ex7_NewtonRhapson(x0)

    f(x) = 1-0.5x^(-0.5)-0.3x^(-0.2)

    h = 10.0^(-15)
    df(x) = (f(x0+h)-f(x0-h))/(2h)

    iter = 1
    path = [x0 0; x0 f(x0)]
    results = ["iter" "x" "f(x)"; iter x0 f(x0)]

    while iter <=1000 && abs(f(x0))>h
        path = [path; (x0 - f(x0)/df(x0)) f(x0)]
        x0 = x0 - f(x0)/df(x0); iter += 1
        path = [path; x0 f(x0)]
        results = [results; iter x0 f(x0)]
    end

    x = 0.2:0.01:1.4
    p = plot(
        layer(x=[x[1] x[length(x)]], y=[0 0], Geom.line, Theme(default_color=
        layer(x=x, y=f.(x), Geom.line, Theme(default_color=colorant"blue"), o
        layer(x=path[:,1], y=path[:,2], Geom.line(preserve_order=true), Theme
        Coord.Cartesian(xmin=x[1],xmax=x[length(x)])
    )

    return results, [results[1:4,:]; results[(size(results)[1]-2):size(result
end;
```

In [10]:

```
ex7_NewtonRhapson(1)[2]
```

Out[10]:

```
7×3 Array{Any,2}:
   "iter"   "x"        "f(x)"
  1.0      1.0        0.2
  2.0      0.279424  -0.333025
  3.0      0.44607   -0.101199
  9.0      0.567094  -7.9875e-13
 10.0      0.567094   1.14908e-14
 11.0      0.567094   3.88578e-16
```

In [11]:

```
@manipulate for x0 in 0.2:0.1:1.3
    ex7_NewtonRhapson(x0)[3]
end
```

x0 0.7

Out[11]: