

Assignment 2: Optimisation using Newton's method

1. The evaluation of the second derivative is given by

```
eval'' = eval' . derive = eval . derive . derive
```

- i. Show that `eval''` is not a homomorphism.
 - ii. What datatype is needed to have a homomorphism from `Expr` in this case? Show this for the case of multiplication.
 - iii. What datatype is needed for the homomorphism that is analogous to apply? Give instances of the numerical classes (`Num`, `Fractional`, `Float`) for this datatype, as well as an embedding of `Const` and `Id`.
2. Newton's method allows us to find zeros of a large class of functions in a given interval. The following description of Newton's method follows Bird and Wadler (1988), page 23:

```
newton :: (Double -> Double) -> Double -> Double -> Double
newton f eps x = if abs fx < eps
                  then x
                  else if fx' /= 0 then newton f eps next
                           else newton f eps (x + eps)
    where fx          = f x
          fx'         = undefined -- f' x (derivative of f at x)
          next        = x - (fx / fx')
```

Implement Newton's method, using the datatype of 1.iii above for computing the derivatives. In other words, use the code above to implement

```
newton :: (FDD Double -> FDD Double) -> Double -> Double -> Double
```

where `FDD a` is the datatype from 1.iii, in order to obtain the appropriate value for `f' x`.

Test your implementation on the following functions:

```
test0 = x^2           -- one zero, in 0
test1 = x^2 - 1       -- two zeros, in -1 and 1
test2 = sin           -- many, many zeros (n*pi)
test3 n x y = y^n - fddId x -- nth root of x
    where fddId is the embedding of Id
```

For each of these functions, apply Newton's method to a number of starting points from a sensible interval. For example:

```
fmap (newton test1 0.001) [-2.0, -1.5 .. 2.0]
```

but be aware that the method might not always converge!

3. We can find the optima of a twice-differentiable function on an interval by finding the zeros of its derivative on that interval, and checking the second derivative. If x_0 is a zero of f' , then

- if $f'' x_0 < 0$, then x_0 is a maximum
- if $f'' x_0 > 0$, then x_0 is a minimum
- if $f'' x_0 = 0$, then, if $f'' (x_0 - \varepsilon) * f'' (x_0 + \varepsilon) < 0$ (i.e., f'' changes its sign in the neighbourhood of x_0), x_0 is an inflection point (not an optimum)
- otherwise, we don't know

Use Newton's method to find the optima of the test functions from point 2. That is, implement a function

```
optim :: (FDD Double -> FDD Double) -> Double -> Double -> Result Double
```

so that `optim f eps x` uses Newton's method to find a zero of f' starting from x . If y is the result (i.e. $f' y$ is within `eps` of 0), then check the second derivative, returning `Maximum y` if $f'' y < 0$, `Minimum y` if $f'' > y$, and `Dunno y` if $f'' = 0$.

As before, use several starting points.

Hint: you might want to modify the code you've written for Newton's method at point 2.

- *Submission*: Assignments are to be submitted via Fire.
- *Deadline*: Tuesday, 2016-03-01, 23:59.
- *Grading*: Discussions with each of the teams during the exercises session of Thursday, 2016-03-03.

References

Bird, R, and P Wadler. 1988. "Introduction to Functional Programming." Prentice-Hall, Englewood Cliffs, NJ.