

# Math 224: Scientific Computing I

Homework 3.2

Assigned Friday, Sept 10, 2004

Due Friday, Sept 24, 2004

## Newton's Method, Volume 2

-2. Mistakes in Volume 1: problem 4 should have said

4. Add a few lines of code to each of the routines from (3)...

-1. Supplementary reading: Numerical Recipes in C, section 9.4 (available on the web).

0. Get a copy of `complex.hpp` (from the course web page or from `/home/class/mth224-01/source`) or any other equivalent C++ package for doing complex arithmetic. You will need this for problems 2–5.

1. [The logistic map] One iterative map that has been studied extensively to illustrate complicated dynamics coming out of simple systems is the *logistic map*,

$$x_{k+1} = g(x_k), \quad g(x) = rx(1-x),$$

where  $r$  is a positive constant. This problem will introduce you to some of the interesting properties of this map:

(a) For  $0 < x_k < 1$ , the map is positive,  $g(x_k) > 0$ , For what range of values for  $r$  does  $x_{k+1}$  satisfy  $0 \leq x_{k+1} \leq 1$ ?

(b) Find the fixed points  $x_*$  of  $g(x)$  as functions of  $r$ .

(c) Calculate the linearized stability ( $\lambda$ ) of the fixed points as functions of  $r$ .

(d) At a parameter value  $r_c$  where a fixed point changes from being stable to being unstable, this is accompanied by the introduction of new stable solutions. This is called a *bifurcation*. The new solutions are not fixed points of  $g(x)$ , they are periodic solutions. For example a period-2 solution alternates between two values:  $x^a$  and  $x^b$  with  $x^b = g(x^a)$  and  $x^a = g(x^b)$ . These periodic solutions of  $g$  are *fixed points of the iterated map*  $g_2(x) = g(g(x))$ .

Compute the range of stable period-2 oscillations.

Plot the  $x^a, x^b$  values as functions of  $r$  ( $r$  vs.  $x$ ).<sup>1</sup>

Also, compute and plot the range of stable period-4 oscillations,  $g(g(g(g(x))))$ .

This is the famous *period-doubling cascade* that continues to period- $2^\infty$ , followed by more complicated *chaotic dynamics*.

2. Modify one of your real-valued `newton()` routines (either `newton2()` [with the optimal value for  $dx$ ] or `newton3()`) to apply to complex-valued functions

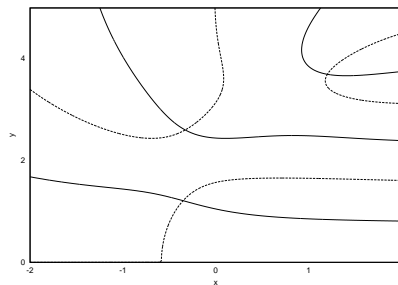
```
complex cnewton(complex (*f)(complex z), complex z0);
```

3. [Harmonic conjugate functions] Find the three intersection points of the two curves

$$u(x, y) = e^{2x} \cos(2y) + \cos x \cosh y + x^2 - y^2$$

$$v(x, y) = e^{2x} \sin(2y) - \sin x \sinh y + 2xy$$

in the domain  $\{-2 \leq x \leq 2, 0 \leq y \leq 5\}$ .



<sup>1</sup>Include the curves for the fixed points of  $g(x)$  in your plot.

- (a) Show that  $u, v$  are harmonic conjugates by showing they satisfy the Cauchy-Riemann equations.
- (b) Find the analytic function  $f(z) = u(x, y) + iv(x, y)$ .
- (c) Use `cnewton()` to find the three intersection points to at least 10 digits of accuracy.  
Hint: The  $x$ -coordinate of one intersection point is  $x_* \approx -0.352194631885$ .

4. [Newton's method and fractals] As described in class, Newton's method will only converge if the initial guess  $z_0$  is "close enough" to the fixed point  $z_*$ . The set of all points  $\{z_0\}$  that converges to  $z_*$  is called the *basin of attraction* of  $z_*$ . Consider the equation

$$f(z) = z^3 + 1 = 0.$$

Find the basin of attraction of the root  $z_* = -1$  for Newton's method.

Follow this outline:

- (a) Use `cnewton()` to generate a finite sequence  $z_k$  starting from an initial value  $z_0$ . Set a condition to determine if the sequence is converging to  $z_*$  or diverging.
- (b) Repeat step (a) starting from each point  $z_0 = x_0 + iy_0$  in the domain  $\{-1 \leq x_0 \leq 0.5, 0 \leq y_0 \leq 1.5\}$ .<sup>2</sup>
- (c) For each run from (b) that converges, output  $x_0$  and  $y_0$  to a file.<sup>3</sup>

Plot the set of points (the basin of attraction).

Note that the boundary of the basin of attraction is a self-similar *fractal*. If you "zoom-in" on any part of it, you will see that it contains an infinite sequence of reduced copies of itself!

5. [The return of the Wilkinson polynomial] Reconsider the Wilkinson polynomial for complex variables,

$$W_\alpha(z) \equiv (z - 1)(z - 2)(z - 3) \cdots (z - 19)(z - 20) - \alpha.$$

You will use `cnewton()` to obtain all the roots as a function of the parameter  $\alpha$ ,  $z_*^m(\alpha)$  for  $m = 1, 2, \dots, 20$ :

- (a) Revise your function `double w(double x);` to complex-variable form.
- (b) Starting from  $\alpha = 10^{10}$ , try an initial guess of  $z_0 = 1$  to converge to the first solution,  $z_*^1(10^{10}) \approx 1$  using `cnewton()`.
- (c) Next, increase  $\alpha$ , `alpha*=1.001` (or smaller), (up to  $\alpha \leq 10^{15}$ ) and try to find the solution at this value of  $\alpha$ . This time, as the initial guess  $z_0$ , use the solution  $z_*$  from the previous value of  $\alpha$  in (b). This is called a *parameter-continuation method*. The idea is that  $z_*^1(\alpha_{prev})$  should be a better of the new solution than just  $z_0 = 1$ .

- (d) But this won't help to get us the complex solutions, will it? Isn't it true that since the polynomial is purely real, and all the solutions up to some point are purely real, we are "stuck" with getting only purely-real iterates  $z_k$  from Newton's method?

Yes, actually that's true, but its easy to fix. Here's how: instead of using  $z_0 = 1$  as the initial guess, or even  $z_0 = z_*^1(\alpha_{prev})$ , try  $z_0 = z_*^1(\alpha_{prev}) \pm i10^{-6}$ . I.e. add a small imaginary perturbation – if the answer really should be real, then this perturbation will decrease under iteration, but if the answer is complex, then this gives us just the right 'push' to get un-stuck and move to complex-valued solutions.

- (e) Plot  $\text{Re}(z_*^1(\alpha))$  and  $\text{Re}(z_*^1(\alpha)) + \text{Im}(z_*^1(\alpha))$  on the same plot, as functions of  $\alpha$  (log- $\alpha$   $x$ -axis).
- (f) Repeat steps (b-e) for all twenty roots. Your answer should include only ONE plot with all twenty roots plotted together. (If this is too messy, split it into a small number of plots.)
- (g) One root should be approximately  $z_*^8(8.00037 \times 10^{12}) \approx 8.353747443625275 - i1.056508655959121$ .  
Find the value  $z_*^7(10^{15})$  (the root starting from  $z_*^7(0) = 7$ ) as accurately as possible.

<sup>2</sup>With  $\Delta x = \Delta y = 0.005$  or smaller.

<sup>3</sup>You should find that  $z_0 = -0.462011 + i0.522831$  does not converge, but  $z_0 = -0.222418 + i0.65408$  does.