Lecture 10, Sci. Comp. for DPhil Students

Nick Trefethen, Tuesday 20.11.18

Today

- III.1 Newton's method for a single equation
- III.2 Newton's method for a system of equations

Handouts

- Quiz 5
- Assignment 3 solutions
- Assignment 4
- `m18_prettynewton.m`
  `m19_newtona.m, m19_newtonb.m, m19_newtonc.m`
  `m20_newtonsystem.m`
  `m21_newtonsystemChebfun2.m`

Announcements

- Assignment 3 due now
- Assignment 4 due 336 hours from now at Andrew Wiles reception
- Pass around Nocedal & Wright

We've finished with linear algebra and a Chebfun demo, and now for the final 3 lectures of this first term, we will talk about optimization. This subject keeps growing in importance.

Very interesting talk yesterday evening

Chris Bishop of Microsoft on "The Mathematics Behind the AI Revolution"

Classical ideas of continuous mathematics and numerical computation are becoming more central in computer science

[screen shot]

III OPTIMIZATION

Optimization = minimization (& maximization) / zerofinding

```
...typically for functions of several variables

...often subject to equality or inequality constraints
```

Outstanding textbook: Nocedal & Wright, *Numerical Optimization*, 2nd ed., 2006 (see our Web page)

Our focus:

- continuous (not discrete)
- deterministic (not stochastic)
- medium scale, high-ish accuracy (not machine learning, data science)

III.1 Newton's method for a single equation

Given: function $F(x)$, typically nonlinear.

Goal: find a **zero** or **root** x^* of F, i.e., $F(x^*) = 0$.

I presume you all know **Newton's method**:

```
 -------------------------------
|                               |
|  Given initial guess  x       |
|                        0      |
|                               |
|  For k = 0, 1, ...            |
|                               |
|                               |
|      s  =  - F(x ) / F'(x )   |          s  stands for "step"
|       k         k        k    |
|                               |
|      x    = x  + s            |
|       k+1    k    k           |
|                               |
 -------------------------------
```

If F is twice differentiable and $F'(x^*) \neq 0$, then if x_0 is sufficiently close to x^* the convergence is **quadratic**—i.e., the number of correct digits asymptotically doubles at each step.

[m18_prettynewton.m]

[m19_newtona.m/m19_newtonb.m/m19_newtonc.m]

III.2 Newton's method for a system of equations

Consider now $F : R^n \to R^n$. Seek $x^* \in R^n$ s.t. $F(x^*) = 0$. I.E., we have a system of n eqs in n unknowns:

$$F_1(x_1, \ldots, x_n) = 0$$

$$\vdots$$

$$F_n(x_1, \ldots, x_n) = 0$$

There's an exactly analogous Newton method. To state it we need to define the derivative of A.

Definition. Given $F : R^n \to R^m$, the **derivative** of F at $x \in R^n$ is the $m \times n$ **Jacobian matrix** defined by

$$[J(x)]_{ij} = [F'(x)]_{ij} = \frac{\partial F_i}{\partial x_j}(x)$$

Example. $F : R^2 \to R^3$.

$$F_1(x_1, x_2) = x_1^2, \quad F_2(x_1, x_2) = x_1 + x_1 x_2, \quad F_3(x_1, x_2) = x_1 \exp(x_2).$$

```
                |  2x         0      |
                |    1               |
                |                    |
       F'(x) =  |                    |
                | 1+x         x      |
                |    2         1     |
                |                    |
                |                    |
                |exp(x )  x exp(x )|
                |     2    1      2 |
```

Motivation:

$$F(x + \Delta x) \approx F(x) + F'(x)\Delta x$$

```
      |   |       \       |
     n-vectors    m-vectors
```

...with equality in limit $\Delta x \to 0$

Newton's method

```
 -----------------------------------
```

```
|                                     |
|  Given initial guess  x             |
|                        0            |
|                                     |
|  For k = 0, 1, ...                  |
|                                     |
|     Evaluate F(x ) and F'(x )       |    (here n=m so F' is square)
|                 k          k        |
|                                     |
|     Solve F'(x )s  = -F(x ) for s  |
|               k  k       k       k |
|                                     |
|     x    = x  + s                   |
|      k+1    k    k                  |
|                                     |
 ------------------------------------
```

Again, quadratic convergence under suitable hypotheses:

F twice differentiable, $F'(x^*)$ nonsingular, x_0 close enough to x^*.

In MATLAB: `fzero` for a scalar problem, `fsolve` for a system
(in the Optimization Toolbox)

These do much more than just Newton's method: they are much more robust.

For an example, consider

$$F(x, y) = \begin{pmatrix} \sin(x + y) - e^{-x^2} \\ 3x - xy^2 - 1 \end{pmatrix}$$

with Jacobian

$$J(x, y) = \begin{pmatrix} \cos(x + y) + 2xe^{-x^2} & \cos(x + y) \\ 3 - y^2 & -2xy \end{pmatrix}$$

`[ m20_newtonsystem.m ]`

`[ m21_newtonsystemChebfun2.m ]`