conjgrad

May 18, 2018

Conjugate Gradient by Iliya Sabzevari

The conjugate gradient algorithm is an iterative way to solve a symmetric, positive-definite linear system of equations. I would give a slight synopsis of the general idea but I think wikipedia already does an excellent job. The program can be seen below.

```
In [1]: import numpy as np
        import math

        #Solves Ax = b for x, where A is a real,symmetric, positive-definite matrix
        #for more infomration about the conjugate gradient algorithm, wikipedia has an excellent

        def conjgrad(Hessian_func,b,xguess):
                x0 = xguess
                r0 = b - Hessian_func(x0)
                p0 = r0
                dotr0 = np.dot(r0.T,r0)
                while True:
                        Ap = Hessian_func(p0)
                        pAp = np.dot(p0.T,Ap)
                        a = dotr0/pAp
                        x1 = x0 + a*p0
                        r1 = r0 - a*Ap
                        dotr1 = np.dot(r1.T,r1)
                        if math.sqrt(dotr1) < 1e-9:
                                break
                        b = dotr1/dotr0
                        p1 = r1 + b*p0

                        x0 = x1
                        r0 = r1
                        p0 = p1
                        dotr0 = dotr1
                return x1
```

I tested the algorithm with the example given in wikipedia, you can see for yourself it works.

```
In [3]: #!/usr/bin/env python
        import numpy as np
```

```
        import conjgrad as cj

        b = np.array([1,2],dtype = np.float64)
        xguess = np.array([0,0],dtype = np.float64)

        def Hessian(v):
                A = np.array([[4,1],[1,3]],dtype = np.float64)
                return np.dot(A,v)


        x = cj.conjgrad(Hessian,b,xguess)

        print(x)
```

```
[0.09090909 0.63636364]
```