TFY4235/FY8904 Computational Physics, Spring 2013

Solution Set 10

Problem 1.

Here is a simple program that finds the maximum of $f(x) = e^{-x^2} \cos x$ on the interval $[-5\pi, 5\pi]$ through evolution of a population of 100 individuals (bit strungs) — called `in(ipop,i)` in the program. In order not to make the program unreadable, I have not packed the bit strings into words. Rather, every element `i` in `in(ipop,i)` is a bit.
Genetic algorithms consist of three main elements: (1) Selection of most fit individuals. I do this through comparing randomly chosen pairs and with probability `pvlg` chossing that with largest f value for inclusion in next generation. (2) Cross linking: With probability `qvlg` we cut both individuals of a randomly chosen pair in two and stitches them together again with their "tails" exchanged (see the program for details). (3) With probability `rmut` we turn the value of the bits around. This is the "mutation" step.

```
      program genalg
c Genetic algorithm for finding the maximum of exp(-x**2) cos x
c on the interval [-5pi,5pi]
      parameter(npop=100,lstr=30,
      c ngen=100,rmut=0.02,pvlg=0.75,qvlg=0.50)
c lstr < 32
      dimension in(npop,lstr),nn(npop,lstr),x(npop),f(npop)
      ibm=4711
      do i=1,1000
      ibm=ibm*16807
      enddo
      rinv=0.5/(2.**31-1.)
      pi =4.*atan(1.)
      pi5=pi*5.
      dvl=10.*pi/2.**lstr
      amut=0.5+rmut
      avlg=0.5+pvlg
      bvlg=0.5+qvlg
      do ipop=1,npop
      do i=1,lstr
      ibm=ibm*16807
      iran=int(rinv*ibm+1.)
      in(ipop,i)=iran
      enddo
      enddo
```

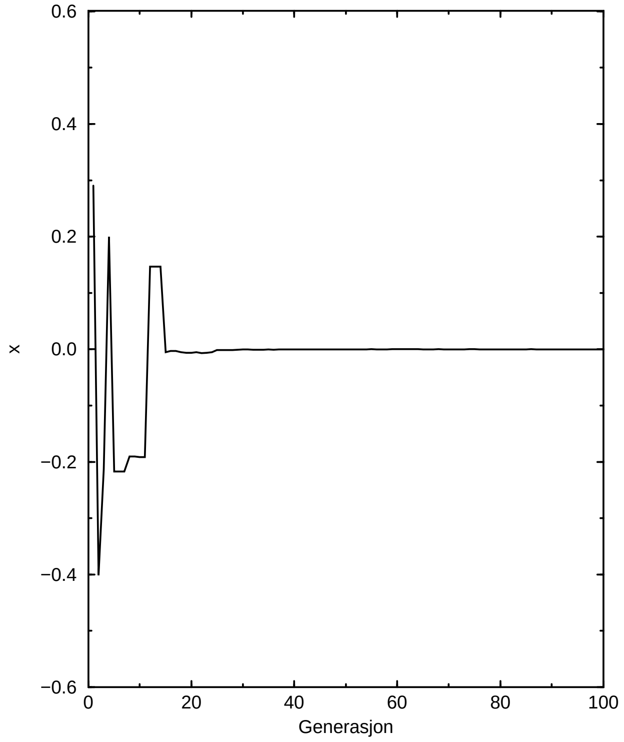
```
      do igen=1,ngen
      fmax=-1.e8
      xmax=0.
      do ipop=1,npop
      ig=0
      do i=1,lstr
      ig=ig+2**(i-1)*in(ipop,i)
      enddo
      x(ipop)=dvl*ig-pi5
      f(ipop)=exp(-x(ipop)**2)*cos(x(ipop))
      if(f(ipop).gt.fmax) then
      fmax=f(ipop)
      xmax=x(ipop)
      endif
      enddo
      write(*,*) igen,xmax,fmax
c selection
      do ipop=1,npop
      ibm=ibm*16807
      jpop=npop*(rinv*float(ibm)+0.5)+1
      ibm=ibm*16807
      kpop=npop*(rinv*float(ibm)+0.5)+1
      imin=jpop
      imax=kpop
      if(f(jpop).gt.f(kpop)) then
      imin=kpop
      imax=jpop
      endif
      ibm=ibm*16807
      ivlg=int(avlg-rinv*ibm)
      if(ivlg.eq.1) then
      do i=1,lstr
      nn(ipop,i)=in(imax,i)
      enddo
      else
      do i=1,lstr
      nn(ipop,i)=in(imin,i)
      enddo
      endif
      enddo
      do ipop=1,npop
      do i=1,lstr
      in(ipop,i)=nn(ipop,i)
      enddo
```

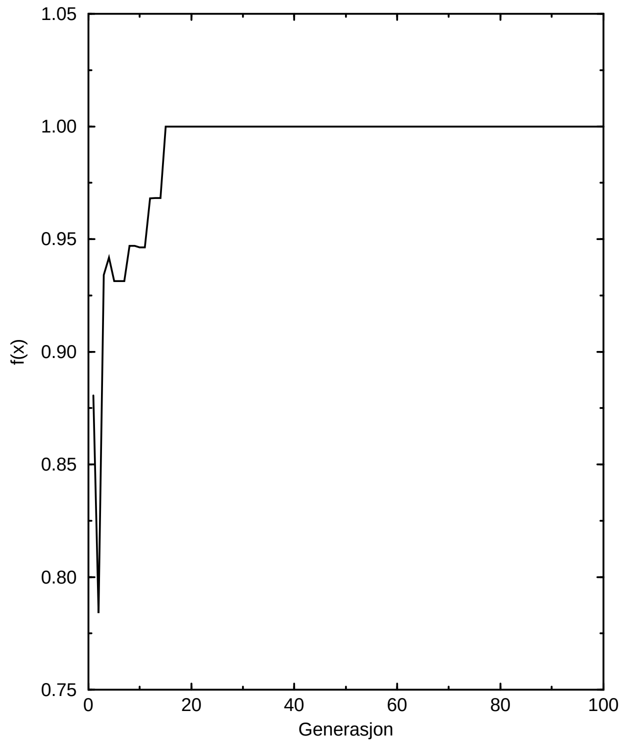
```
      enddo
c crossover
      do ipop=1,npop
      ibm=ibm*16807
      ivlg=int(bvlg-rinv*ibm)
      if(ivlg.eq.1) then
      ibm=ibm*16807
      kpop=npop*(rinv*float(ibm)+0.5)+1
      ibm=ibm*16807
      kstr=lstr*(rinv*float(ibm)+0.5)+1
      do k=1,kstr
      nn(kpop,k)=in(kpop,k)
      enddo
      do k=1,kstr
      in(kpop,k)=in(ipop,k)
      enddo
      do k=1,kstr
      in(ipop,k)=nn(kpop,k)
      nn(kpop,k)=0
      enddo
      endif
      enddo
c mutation
      do ipop=1,npop
      do i=1,lstr
      ibm=ibm*16807
      imut=int(amut-rinv*ibm)
      in(ipop,i)=and(xor(in(ipop,i),imut),1)
      enddo
      enddo
      enddo
      end
```

(Note that the program uses two bit operations. These are not portable. Syntax varies from compiler to compiler.)

The paper by R. L. Riolo, "Survival of the fittest bit," (Scientific American, page 89, July 1992) describes in detail how to write a genetic algorithm.

In the figures below I show the development of the solution x of the equation $f(x) = \max_{x'} f(x')$ (which is $x = 0$), and the development of f itself — both as functions of the number of generations. After 100 generations, $x = -5.53131E - 05$.


0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
x
0
20
40
60
80
100
Generasjon


1.05
1.00
0.95
0.90
0.85
0.80
0.75
f(x)
0
20
40
60
80
100
Generasjon