Complexity in social dynamics : from the micro to the macro
Laboratory 4

Franco Bagnoli

Namur 7-18/4/2008

1 Laboratory 4

Evolution and game theory

1. Evolution on a flat landscape: mutations.
2. Evolution on a smooth fitness landscape: quasispecies and the Red Queen.
3. Evolution of a sharp landscape: the error threshold.
4. Niches and coexistence in firness landscapes.
5. Competition: a stabilizing force.
6. Evolution of cooperation: direct reciprocity.

Evolutionary models

- An individual is modeled as a vector of L genes (that take value 0 and 1).
- The phenotype f is just the sum of the genes ($0 \leq f \leq L$).
- The firness is a function of the phenotype and the phenotype distribution of other individuals (for competition).
- For modeling an evolutionary population (fixed size:`quasispecies.f90`), we just compare the fitness of two individuals. That with lower fitness tends to disappear, replaced by a copy of the opponent, with eveltual mutations.

`quasispecies.f90`

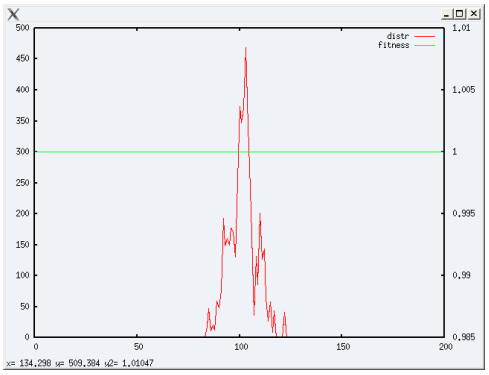
```
module fit
contains
  function pheno(g)
    integer ::pheno
    integer*1 :: g(:)
    integer :: L,i

    L = size(g)
    pheno = 0
    do i=1,L
      pheno = pheno + g(i)
    end do
  end function

  function fitness(f, fd)
    implicit none
    real :: fitness
    integer :: f ! phenotype
    integer :: fd(0:) ! phenotype distribution
    integer :: LL, i
    real :: d,h
    integer :: NN

    LL = size(fd)
    LL = LL-1
    NN = sum(fd)

    !fitness=1 !  neutral evolution

    ! a sharp-peaked landscape
```

```
    !if (f == 0) then
    !  fitness = 1
    !else
    !  fitness = 0
    !end if
    ! smooth landscape
    !fitness = (real(LL-f)/LL)**.1
    ! smooth landscape
    fitness = abs(real(LL/2-f)/LL)**1.01

    !competition
!     h = 0
!     do i=0,LL
!       d = abs(i-f) ! phenotypic distance
!       h = h - 1*fd(i)*exp(-.01*d)/NN
!     end do
!     fitness = exp(h)
  end function
end module

program quasispecies
  use fit
  implicit none
  integer, parameter :: N=5000 !population size
  integer, parameter :: L=200 ! genome size
  integer :: i, j, k,kk, t
  integer :: TMAX=10000
  integer*1 :: x(N,L) !population
  real :: A(N) ! fitness
  integer :: f(N) ! phenotype
  integer :: fd(0:L) ! phenotype distribution
  real :: r , AA
  real :: mu=0.0001 !mutation prob per gene
  real :: Temp=0.1
  character*200 :: str


  call random_seed()
  !initialization
  fd = 0
  do i=1, N
    do j=1, L
      call random_number(r)
      x(i,j) = floor(r+0.5)
    end do
    !x(i,:) = 0
    f(i) = pheno(x(i,:))
    fd(f(i)) = fd(f(i))+1
   end do

  ! compute fitness
  do i=1, N
    A(i) = fitness(f(i), fd)
  end do

  call gnuplotOpen("gnuplot")
```

```
  call gnuplotExecute("set term x11; set mouse;"//&
      "set ytics nomirror; set y2tics")


  ! evolution
  do t=1, TMAX
    do kk=1, N ! a MonteCarlo step
      ! pick two individuals at random
      call random_number(r)
      i = floor(r*N)+1
      call random_number(r)
      j = floor(r*N)+1
      ! selection
      call random_number(r)
      if (r < 1/(1+exp((A(i)-A(j))/Temp))) then
        k=i; i=j; j=k ! exchange i and j
      end if
      ! i duplicates into j
      fd(f(j)) = fd(f(j)) -1 ! remove j
      ! clone i into j
      x(j,:) = x(i,:)
      !mutations: up to 1 mutation at random
      call random_number(r)
      if (r< mu*L) then
        call random_number(r)
        k = floor(r*L)+1
        x(j,k) = 1-x(j,k)
      end if
      ! compute phenotype and fitness
      f(j) = pheno(x(j,:))
      fd(f(j)) = fd(f(j)) +1
      A(j) = fitness(f(j), fd)
    end do
    AA = sum(A)/N ! average fitness
    print *, AA
    !plot
    call gnuplotExecute("plot '-' t 'distr' w l, "//&
        "'-' axis x1y2 t 'fitness' w l")
    do i=0,L
      write(str, *) i, fd(i)
      call gnuplotExecute(str)
    end do
    call gnuplotExecute("end")
    do i=0,L
      write(str, *) i, fitness(i, fd)
      call gnuplotExecute(str)
    end do
    call gnuplotExecute("end")
    call gnuplotFlush()
  end do
end program
```

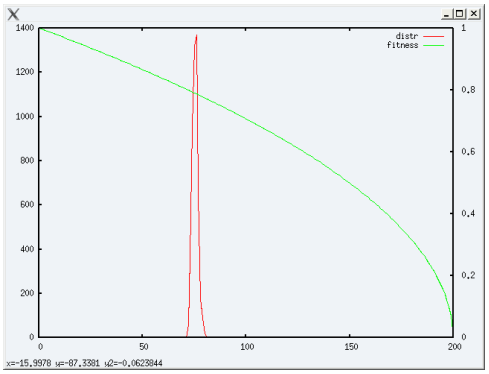
Evolution on a flat landscape

- Witout selection (flat fitness landscape), mutations (random drift) tend to favor the intermediate phenotype (0.5). The asymptotic distribution approximates a binomial one.
- Notice that for a genome length L sufficiently high, the binomial distibution is so sharp that, for finite populations, the expreme values (say, genotype $0, 0, 0, \ldots$ never appear.



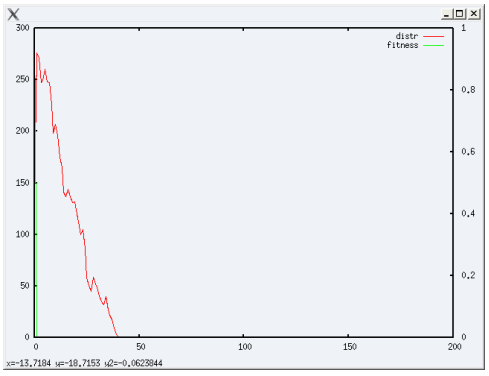
Evolution on smooth landscapes

- In smooth landscapes, there is a competition between fitness (order) and mutations (disorder).
- you can try to explore the relationship between fitness shape, mutations and position and width of quasispecies.



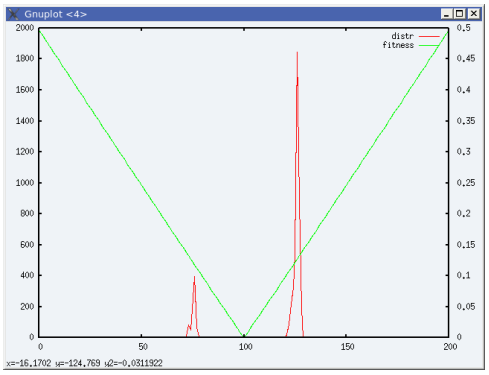
Evolution on sharp landscapes: error threshold

- For a sharp landscape and finite populations, the asymptotic population is a quasispecies centered around the master sequence (here the sequence $0, 0, \ldots$.
- it may happen that the master sequence is lost.
- Since the fitness landscape is flat (except for the master sequence), the evolution is here just a random search of a point in a high-dimensional space: no hope of finding it.



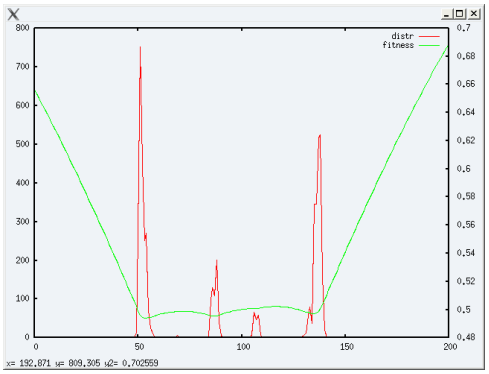
Speciation and coexistence on smooth landscapes

- The allopatric speciation theory identifies speciations with the "discovering" of niches.
- Since niches (fitness maxima) are separated by valleys, one needs "hopeful monsters" that accumulates mutations. This is easier for smaller populations, and therefore in isolated islands.
- However, coexistence is fragile: random fluctuations may bring species into extinction.



Competition

- Competition arises when one individual uses some resource correlated to its phenotype (example: seeds of a diameter related to its beak size).
- In principle, one should simulates at least two species (a prey and a predator), but we can use an "effective" competition term.
- In the presence of competition, more species can occupy the same niche, even is random drift "pushes" phnotypes towards the internediate one.



Evolutionary game theory

- In evolutionary game theory, the fitness landscape is replaced by direct integractions among individuals.
- We simulate here the evolution of cooperation by direct reciprocity (`cooperation.f90`). A population composed by TIT-FOR-TAT (0) and ALL-D (1) strategies is engagen in an round-robin tornament, and accumulate payoff.
- After that, selection takes part, as in `quasispecies.f90`.

cooperation.f90

```
program cooperation
  implicit none
  integer, parameter :: N=200 !population size
  integer :: i, j,ii, k,kk, t, ll
  integer :: TMAX=10
  integer*1 :: x(N) !population 1=tit for tat, 2=allD
  real :: A(N) ! fitness:average payoff
  real :: r , AA, xx
  real :: Temp=0.001 !selection temperature
  real :: w = 0.5 ! end tournament probability
  character*200 :: str
  real :: payoff(0:1,0:1)
  integer :: D1, D2, DD1
  real :: b = 5 ! benefit
  real :: c = 2.5 ! cost
  real :: length, ww
  real :: x0=.01 ! initial fraction of defectors

  payoff(0,0) = b-c ! CC cooperation
  payoff(0,1) = -c ! CD
  payoff(1,0) = b ! DC exploitation
  payoff(1,1) = 0 ! DD

  call random_seed()

  !initialization
  do i=1, N
    call random_number(r)
    x(i)=floor(r+x0)
  end do

  xx = 0
  do i=1, N
    xx = xx + x(i)
  end do
  xx = xx/N
  print *, "initially x=", xx

  ! evolution
  do t=1, TMAX
    A = 0
    length=0
    do k=1, N
      do kk=1, N ! complete tournament
        ! pick opponent at random
        !call random_number(r)
        !j = floor(r*N)+1
        i = k
        j = kk
        ! tournament
        ! first deal of TFT is cooperate, ALLD always defects
        D1 = x(i)
        D2 = x(j)
        A(i) = A(i) + payoff(D1,D2)
        A(j) = A(j) + payoff(D2,D1)
```

```
        ll = 1
!        print *, "------"
!        print *, ll,")", i, "(", x(i), ") plays ",d1, " and scores ", payoff(D1,D2)
!        print *, "vs ", j, "(", x(j), ") plays ",d2, " and scores ", payoff(D2,D1)
        do while (.true.)
          call random_number(r)
          if (r < w) exit !end tournament
          ll = ll + 1
          if (x(i) == 0) then
            DD1 = D2 ! TFT repeats last move of opponent
          else
            DD1 = 1
          end if
          if (x(j) == 0) then
            D2 = D1
          else
            D2 = 1
          end if
          D1 = DD1
!          print *, ll,")", i, "(", x(i), ") plays ",d1, " and scores ", payoff(D1,D2)
!          print *, "vs ", j, "(", x(j), ") plays ",d2, " and scores ", payoff(D2,D1)

          A(i) = A(i) + payoff(D1,D2)
          A(j) = A(j) + payoff(D2,D1)
        end do
        length=length + ll
      end do
    end do
    length = length / N**2
    ww=1/length
    ! selection
    do k=1, N
      i = k
      ! pick an opponent at random
      call random_number(r)
      j = floor(r*N)+1
!      print *, i, a(i),"vs ",  j, a(j)
      ! and select
      call random_number(r)
      if (r < 1/(1+exp((A(i)-A(j))/Temp))) then
        ii=i; i=j; j=ii ! exchange i and j
      end if
!      print *, i, x(i), " wins ", j, x(j)
      ! clone i into j
      x(j) = x(i)
    end do
    AA = sum(A) /N
    xx = 0
    do i=1, N
      xx = xx + x(i)
    end do
!    print *, t,xx/N, c/b, ww, ww/(2-ww),ww/(3-2*ww)
  end do
  print *, "after ", tmax, "rounds, x=",xx/N," c/b=",  c/b, &
     "; ESS (w)=", ww, "; RD (w/(2-w))=", ww/(2-ww),&
     "; AD (w/(3-2w))=", ww/(3-2*ww)
```

```
end program
```

Direct reciprocity

- Depending on the ration c/b of cost with respect to benefit, and the expected number of rounds $1/w$, TIT-FOR-TAT (TFT) may be evolutionary stable (ESS: canot be invaded by a single mutant, but can be invaded by a large grupp of defeaters), robust (RD: a random initial condition with a mall majority of TFF leads to an homogeneous population) or advantageous (AD: even in small population, a fraction of mutatns larger than $1/N$ cannot overcome).
- Try to compare the results with the mean-field approximation by Novak.