Simulations in Balloux et al 2002

Bo Peng (bpeng@rice.edu)

16th March 2005

1 Introduction

This paper

> F Balloux and J Goudet (2002) Statistical properties of population differentiation estimators under stepwise mutation in a finite island model, *Molecular Biolody,* 11, 771-781

compared the performance of two population differentiation estimators: F_{st} and R_{st}, using forward-based simulation program easyPOP. As part of a testing process, I have re-run some of the simulations in this paper using simuPOP. Here is a brief report.

2 Model Assumptions

2.1 Genotypic structure

- 12 microsatellite loci
- unlinked (all on different chromosome)
- allele states: 1 - 999

2.2 Mutation

symmetric stepwise mutation model with $\mu = 0.01, 0.001, 0.0001$. The mixed model is not implemented (but can be easily added if necessary)

2.3 Initial population

Totally random alleles from 1 to 999 with equal probability. There is initially no linkage disequlibrium or population differences ($F_{st} = 0$).

2.4 Statistical measurements

F_{st}: strictly according to Weir and Cockerham 1984.

- one locus measurement is the average of F_{st} based on all available alleles.
- multi-loci measurement is the averaged of all 12 loci.

R_{st} is not implemented.

2.5 Demographic and migration models

- 2 demes of 1000 individuals
- 5 demes of 400 individuals
- 20 demes of 100 individuals

with migration following a island model with rate that allow

- $Nm = 0.1$, 1 or 10.

3 Results

Similar to table 1 in the table. All values are of the last generation.

n	N	m	μ	Nm	EF_{st}	one locus$\hat{F}_{st}$(sd)	reported	12 loci $\hat{F}_{st}$(sd)	reported)
2	1000	0.0001	0.01	0.1	0.065	0.067(0.029)	0.065	0.069 (0.010)	0.065
2	1000	0.0001	0.001	0.1	0.202	0.195(0.087)	0.201	0.209(0.027)	0.208
2	1000	0.0001	0.0001	0.1	0.434	0.371(0.239)	0.338	0.462(0.062)	0.434
2	1000	0.001	0.01	1	0.031	0.031(0.015)	0.031	0.031(0.004)	0.031
2	1000	0.001	0.001	1	0.075	0.076(0.062)	0.072	0.074(0.013)	0.074
2	1000	0.001	0.0001	1	0.104	0.091(0.093)	0.081	0.098(0.028)	0.096
2	1000	0.01	0.01	10	0.008	0.008 (0.003)	0.008	0.009(0.001)	0.008
2	1000	0.01	0.001	10	0.011	0.011 (0.008)	0.011	0.012(0.003)	0.011
2	1000	0.01	0.0001	10	0.012	0.010(0.012)	0.011	0.012(0.005)	0.012
5	400	0.00025	0.01	0.1	0.122	0.125(0.227)	0.121	0.127(0.007)	0.121
5	400	0.00025	0.001	0.1	0.344	0.353(0.076)	0.342	0.353(0.022)	0.344
5	400	0.00025	0.0001	0.1	0.667	0.576(0.169)	0.561	0.598(0.039)	0.591
5	400	0.0025	0.01	1	0.060	0.061 (0.014)	0.060	0.060(0.004)	0.060
5	400	0.0025	0.001	1	0.128	0.126(0.039)	0.126	0.130(0.116)	0.127
5	400	0.0025	0.0001	1	0.160	0.144(0.081)	0.145	0.161(0.028)	0.160
5	400	0.025	0.01	10	0.014	0.016(0.004)	0.014	0.015(0.001)	0.014
5	400	0.025	0.001	10	0.018	0.019(0.006)	0.018	0.019(0.002)	0.018
5	400	0.025	0.0001	10	0.019	0.021(0.016)	0.018	0.019(0.004)	0.019
20	100	0.001	0.01	0.1	0.255	0.256(0.023)	0.255	0.261(0.006)	0.255
20	100	0.001	0.001	0.1	0.547	0.546(0.051)	0.545	0.555(0.019)	0.546
20	100	0.001	0.0001	0.1	0.680	0.695(0.076)	0.678	0.691(0016)	0.683
20	100	0.01	0.01	1	0.114	0.119(0.013)	0.114	0.117(0.005)	0.114
20	100	0.01	0.001	1	0.175	0.177(0.035)	0.174	0.179(0.010)	0.174
20	100	0.01	0.0001	1	0.188	0.183(0.058)	0.181	0.194(0.016)	0.188
20	100	0.1	0.01	10	0.020	0.022(0.003)	0.020	0.023(0.001)	0.020
20	100	0.1	0.001	10	0.020	0.025(0.005)	0.019	0.025(0.001)	0.020
20	100	0.1	0.0001	10	0.019	0.023(0.008)	0.019	0.025(0.002)	0.020

4 Source code (with lots of comments)

```
# this simulation tries to repeat results from
#
#   Balloux & Goudet ( 2002)
#     Statistical properties of population differentiation
#     estimators under stepwise mutation in a finite island model
#
```

Figure 1: $N = 1000$, $m = 0.0001$, $\mu = 0.01$, using locus 2

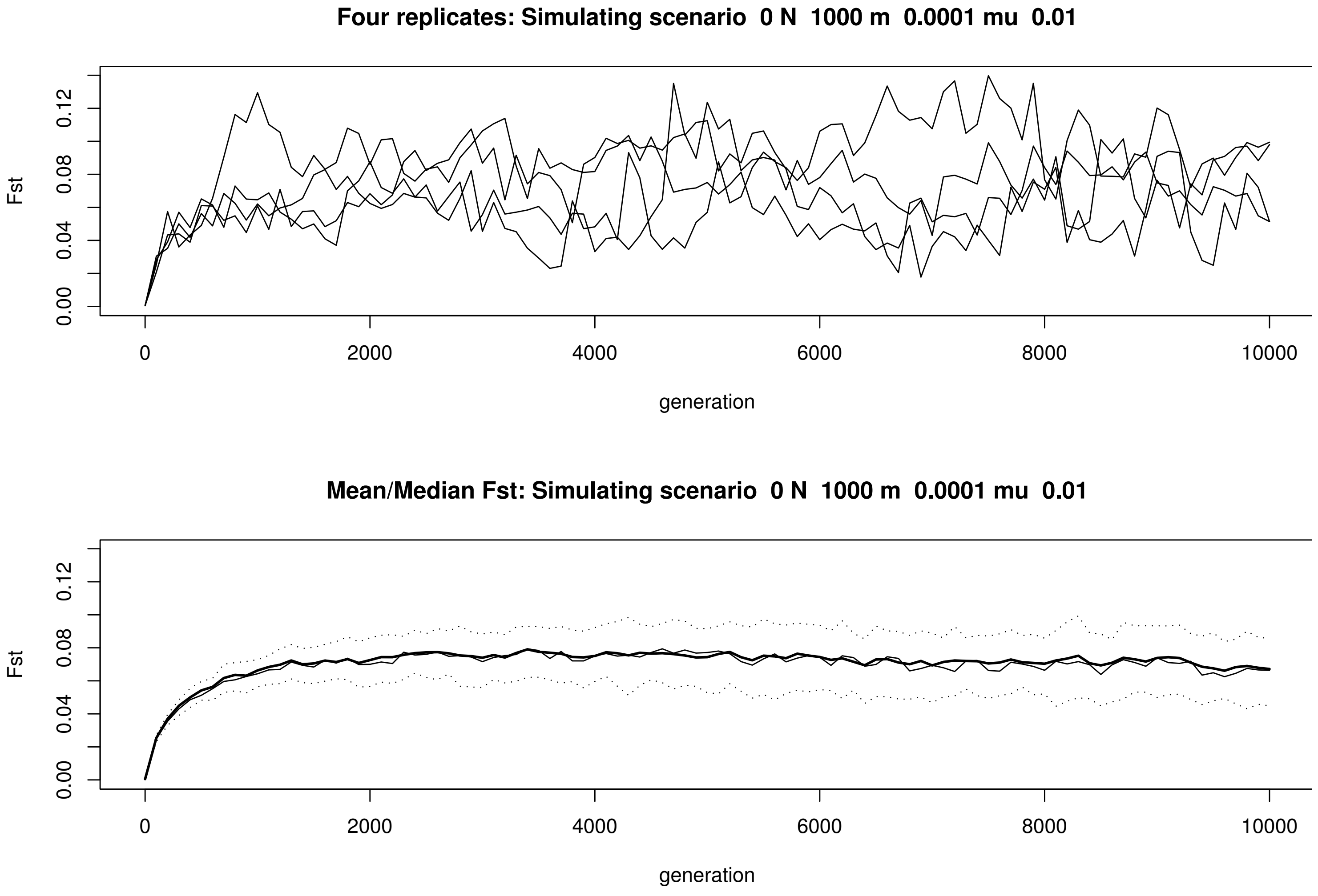
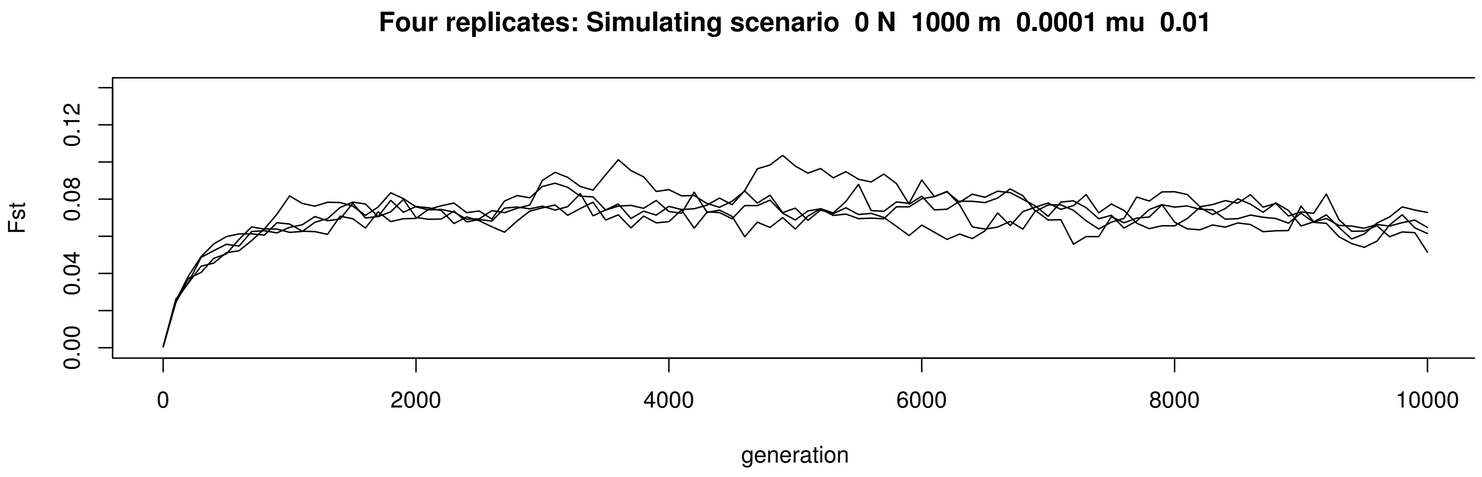


Figure 2: $N = 1000$, $m = 0.0001$, $\mu = 0.01$, using all 12 loci



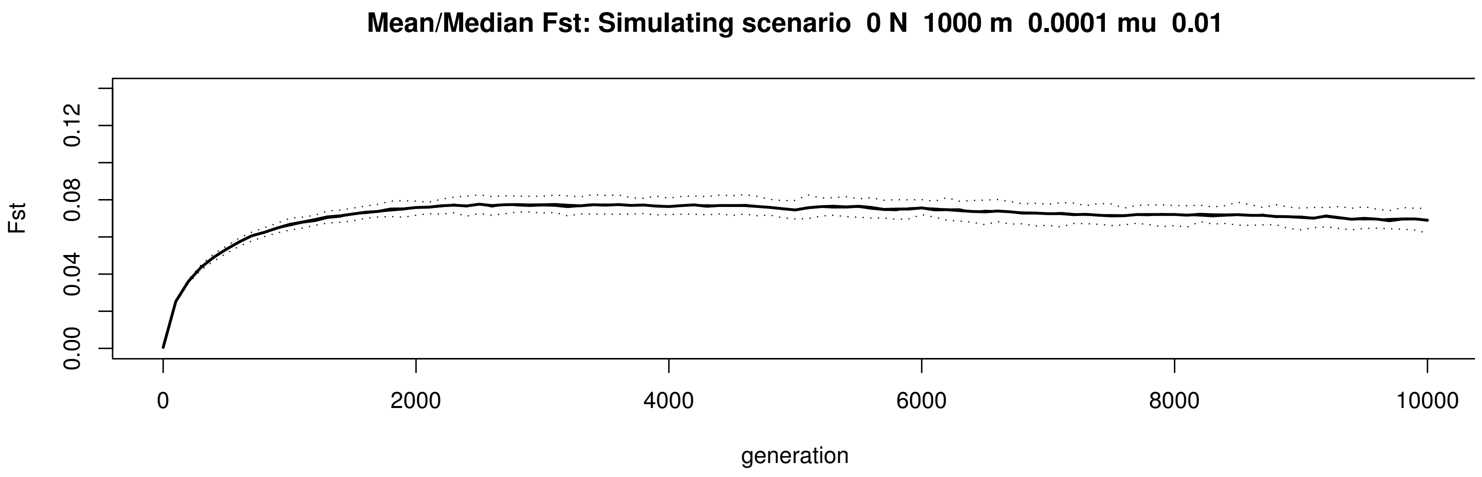


Figure 3: $N = 1000$, $m = 0.0001$, $\mu = 0.001$, using locus 2

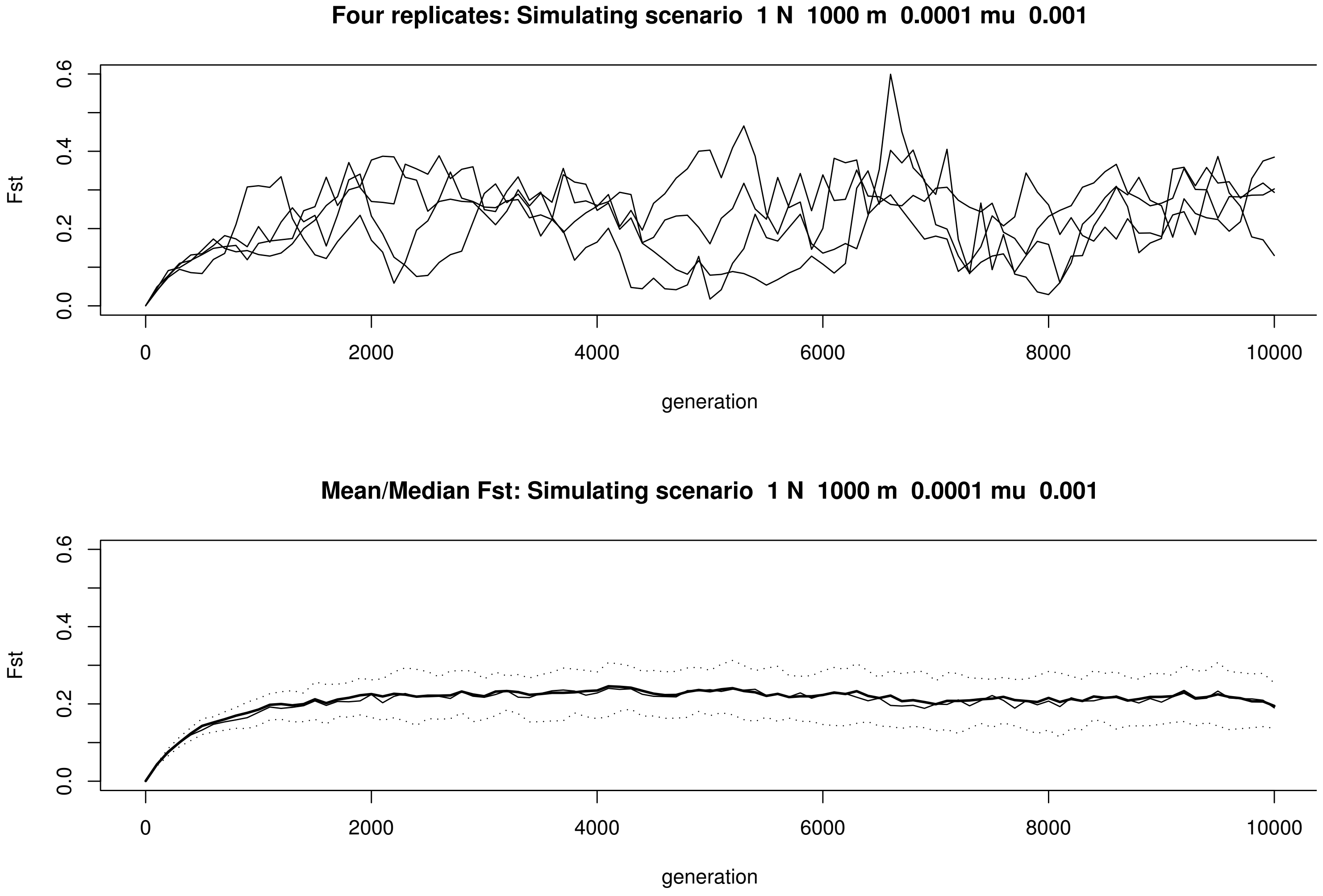
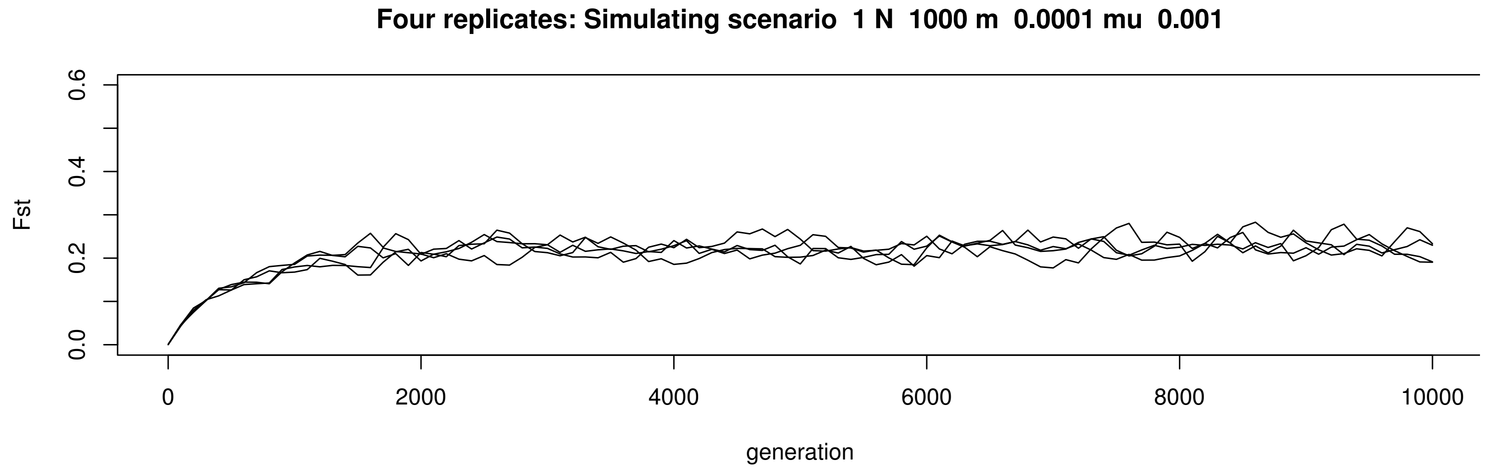


Figure 4: $N = 1000$, $m = 0.0001$, $\mu = 0.001$, using all 12 loci



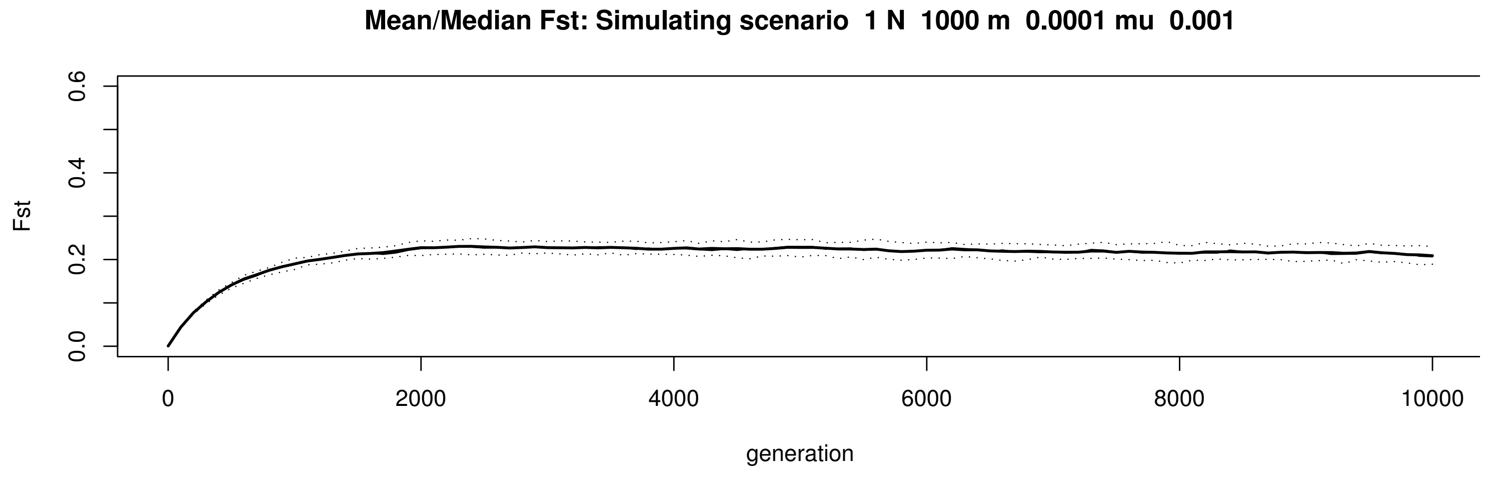


Figure 5: $N = 1000$, $m = 0.0001$, $\mu = 0.0001$, using locus 2

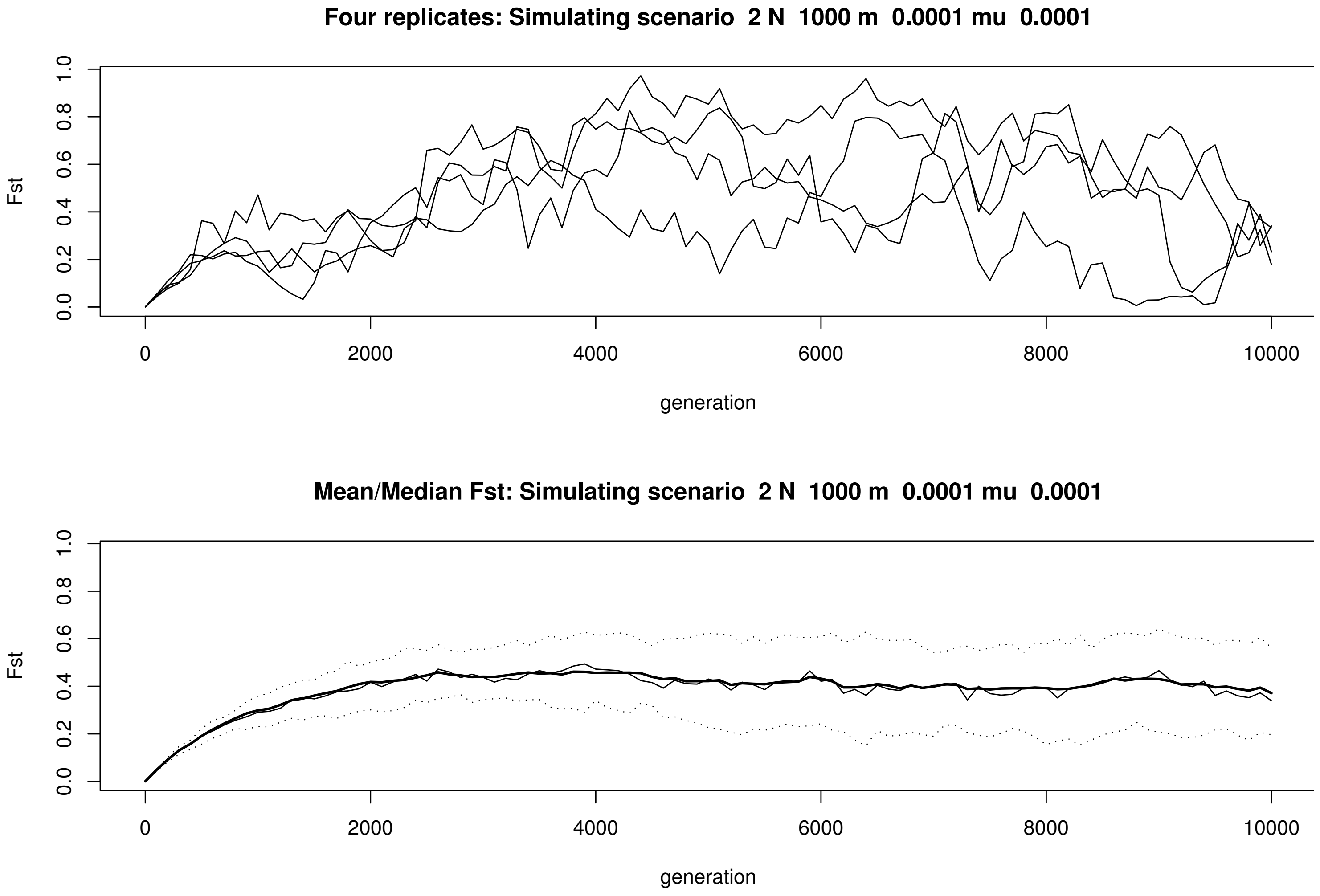
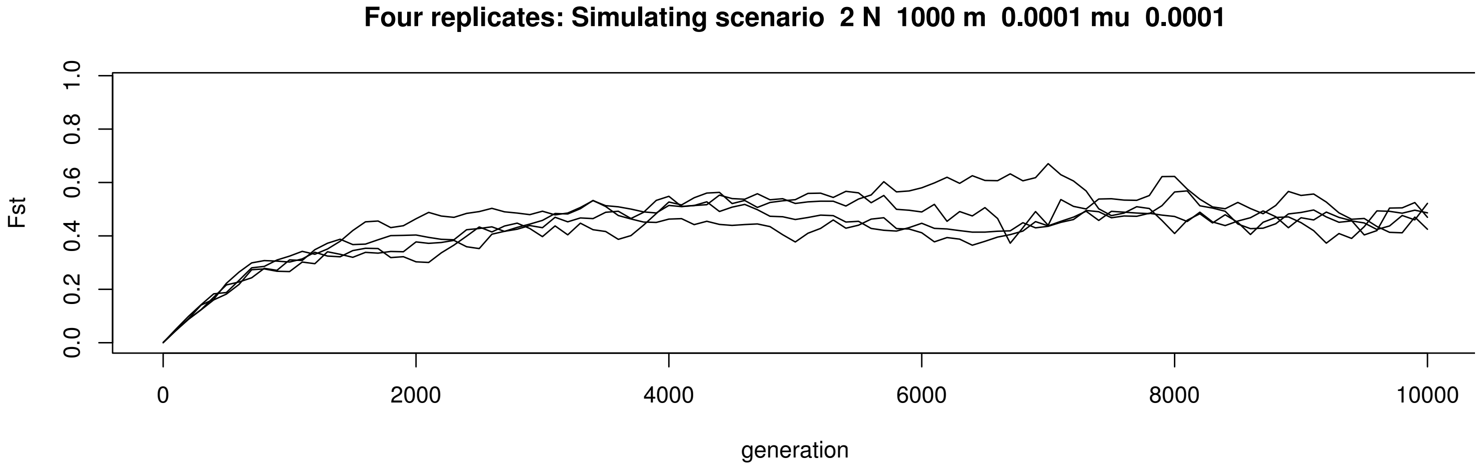
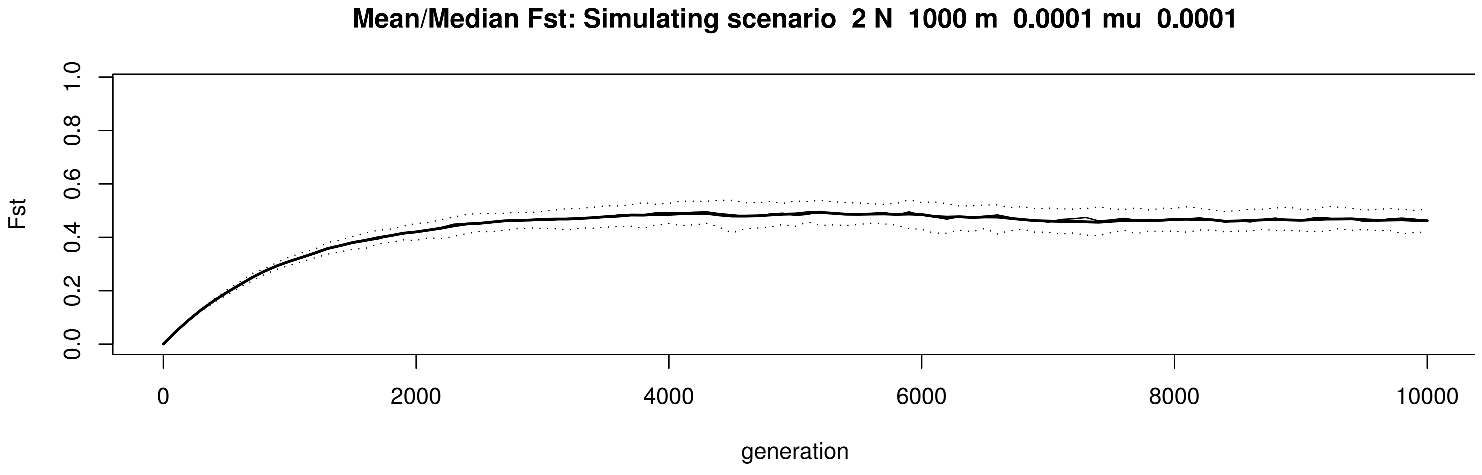


Figure 6: $N = 1000$, $m = 0.0001$, $\mu = 0.0001$, using all 12 loci





```
# should use simuPOP_la in official simuPOP release
from simuPOP import *
from simuUtil import *
#from simuRPy import *

import exceptions

nLoci = 12
ma = 999

# simulation:
# subPop: subpoulation structure
# migrRate: migration rate, actually proportion of migrants
# rep: number of replciates
# endGen: ending generation
#
# return simu for further analysis.
def simulate(subPop, migrRate, mutaRates, nRep=5, endGen=10, visual=[], savePop=False, name=
  numSP = len(subPop)
  if len(mutaRates) != nLoci:
    raise exceptions.ValueError("Please specify mutation rate for each locus.")
  ## initialize (uniform, all possible alleles)
  init =  initByFreq( [1./ma]*ma )
  ## mutation
  ##  different mutation rate to each locus
  ##  step-wise mutation model
  mutate = smmMutator( rates = mutaRates, atLoci = range(0, nLoci),
     maxAllele=ma)
  ## migration, r_ii will be automatically set correctly
  migrate = migrator(
    rates = [[migrRate/(numSP-1)]* numSP] * numSP,
    mode=MigrByProbability)
  ## visualizers
  v1,v2,v3,v4 = noneOp(),noneOp(),noneOp(),noneOp()
  ## visualize all Fst's
  if "Fst" in visual:
    v1 = varPlotter('Fst[:4]', byRep=True, title='Fst', numRep=nRep,
        varDim=4, update=100, win=2000, step=10, saveAs="Fst")
  ## plot average Fst (using all loci)
  if "AvgFst" in visual:
    v2 = varPlotter('AvgFst', byVal=1, title='Fst', numRep=nRep,
        varDim=1, update=10, win=2000, step=10, saveAs="avgFst")
  ## allele frequency
  if "alleleFreq" in visual:
    v3 = varPlotter('[alleleFreq[0][x] for x in range(1,11)]',
        byVal=1, title='expected heterozygosity', numRep=nRep,
        varDim=10, step=10, win=2000,update=100, saveAs="alleleFreq")
  ## heterozygote frequency
  if "heteroFreq" in visual:
    v3 = varPlotter('[heteroFreq[x][0] for x in range(0,4)]',
        byVal=1, title='expected heterozygosity', numRep=nRep,
        varDim=4, step=10, win=3000, update=100, saveAs="heter")
  ## save population for post-morten analysis
  if savePop == True:
    s1 = savePopulation(outputExpr="'" + name + "_%d_%d.bin' % (gen,rep)", begin=5000, step=
  else:
    s1 = noneOp()
  ## calculate Fst and save result (all replicate in one file)
  ## calculate every 50 steps to save time
  stats = stat(Fst=range(0,nLoci), step=50)
  saveFst =  pyEval(r"'%d\t%.6f\t%.6f\t%.6f\t%.6f\t%.6f\t' % (gen,Fst[0],Fst[1],Fst[2],Fst[3
    step = 100, output=">>"+name+"_Fst.txt", name="save Fst values to file")
  saveFst1 = endl(rep=REP_LAST, step=100, output=">>"+name+"_Fst.txt")

  # create population and simulator
  simu = simulator(
    population(subPop=subPop, ploidy=2, loci=[1]*nLoci,
      maxAllele=ma),
    randomMating(newSubPopSize=subPop),
    rep=nRep)
  # start simulation
  simu.evolve(
    preOps = [ init ],
    ops = [
      # report progress
      pyEval(r"'%d\n' % gen", step=100, rep=REP_LAST, name="output generation"),
      mutate, migrate, stats, saveFst, saveFst1,
      s1, v1, v2, v3, v4, ticToc(step=1000,rep=0),
    ],
    end = endGen,
    dryrun = False,
  )
  return simu

## First run, try to get some figures
if False:
  simu = simulate(subPop=[1000]*2, migrRate=0.0001,
    mutaRates=[.0001]*nLoci, endGen=100, nRep=4,
    visual=["Fst","AvgFst"], savePop=False, name='pop')

## verify Fst results, output to fstat format and compare
## The results are identical (on Fst)
```

```
if False:
  pop = LoadPopulation("pop1476_2.bin")
  Stat(pop, Fst=range(0,nLoci))
  print pop.dvars().Fst
  print pop.dvars().AvgFst
  SaveFstat(pop, "pop1476_2.dat")


## run 99 replicates, without visualizer
if True:
  subPops = [ [1000]*2, [400]*5, [100]*20]
  Nm = [0.1, 1., 10.]
  mutaRates = [0.01, 0.001, 0.0001]
  index = 0
  rec = open("simulation.log", "w")
  rec.write("scenario\tsubPopSize\tmigrRate\tmutaRate\n")
  for subPop in subPops:
    for nm in Nm:
      for mutaRate in mutaRates:
        migrRate = nm/subPop[0]
        name = "sce" + str(index)
        print "Simulating scenario ", index, "N ", subPop[0], "m ", migrRate, "mu ", mutaRat
        rec.write("%s\t%d\t%.5f\t%.5f\n" % (name, subPop[0], migrRate, mutaRate))
        if index >= 21:  # starting from # scenario
          simu = simulate(subPop=subPop, migrRate=migrRate,
            mutaRates=[mutaRate]*nLoci, endGen=10000, nRep=100,
            visual=[], savePop=True, name=name)
        index += 1
  rec.close()

##
if False:
  simu = simulate(subPop=[1000]*2, migrRate=0.0001,
    mutaRates=[.0001]*nLoci, endGen=100, nRep=4,
    visual=[], savePop=False, name='pop')
```