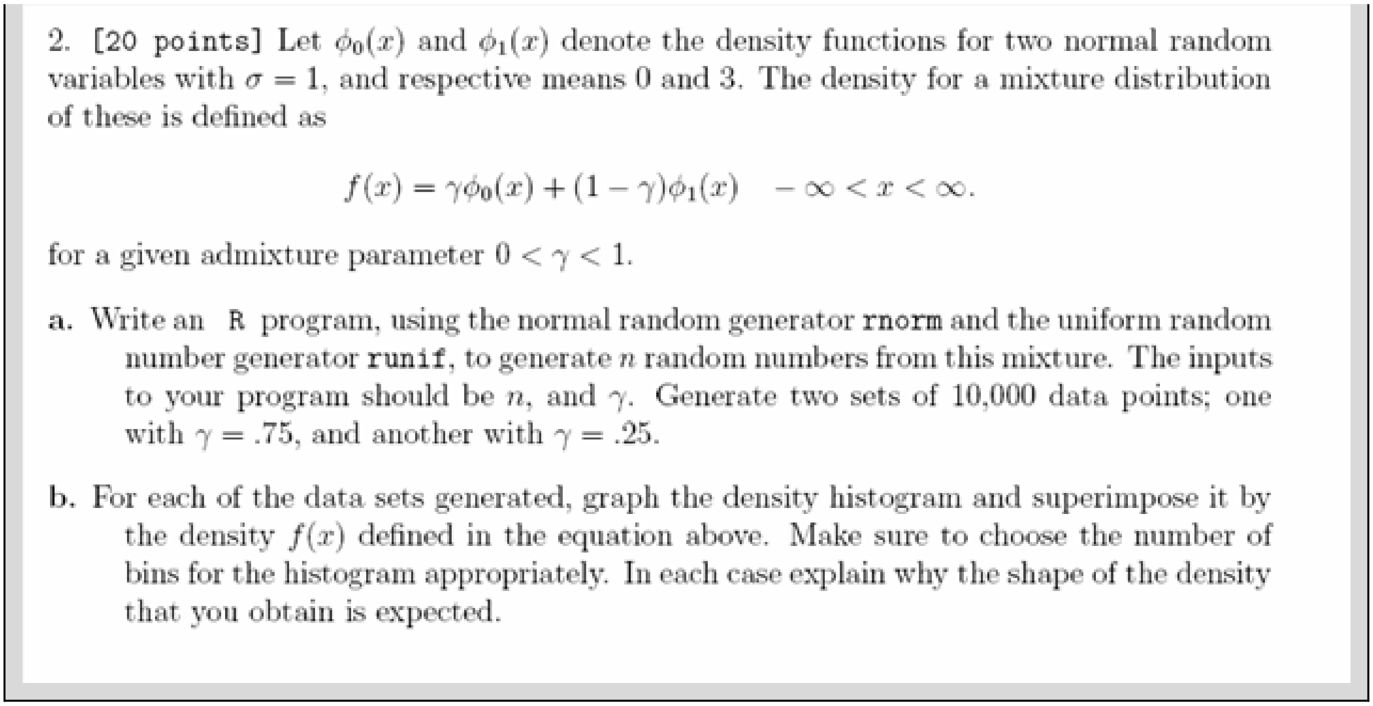
Project one. Problem 2. Mathematics 502 Probability and Statistics

Nasser Abbasi, September 26,2007. California State University, Fullerton



2. [20 points] Let $\phi_0(x)$ and $\phi_1(x)$ denote the density functions for two normal random variables with $\sigma = 1$, and respective means 0 and 3. The density for a mixture distribution of these is defined as

$$f(x) = \gamma\phi_0(x) + (1-\gamma)\phi_1(x) \quad -\infty < x < \infty.$$

for a given admixture parameter $0 < \gamma < 1$.

a. Write an R program, using the normal random generator rnorm and the uniform random number generator runif, to generate n random numbers from this mixture. The inputs to your program should be n, and γ. Generate two sets of 10,000 data points; one with $\gamma = .75$, and another with $\gamma = .25$.

b. For each of the data sets generated, graph the density histogram and superimpose it by the density $f(x)$ defined in the equation above. Make sure to choose the number of bins for the histogram appropriately. In each case explain why the shape of the density that you obtain is expected.

Part 2 (a)

The mixed distribution is

$$f(x) = \gamma\phi_0(x) + (1-\gamma)\,\phi_1(x) \quad -\infty \le x \le \infty$$

Where ϕ_0 is the density function for normal distribution with σ=1,μ=0, and ϕ_1 is the density function for normal distribution with σ=1,μ=3

We need to generate random numbers from the above density function.

The following is the idea of how to solve this problem. Let us consider the case for γ=75% .Generate a r.v. from a uniform distribution, which will be between [0, 1]. Let this number be called ζ. If $\zeta < .76$ then now we will generate a random number from the above $\phi_0(x)$ normal distribution otherwise we will generate a random number from $\phi_1(x)$ distribution. Hence this is the algorithm

Algorithm

Input: γ,σ_0, μ_0, σ_1, μ_1, n (where n is number of random number to generate for mixture $f(x)$
output: n random numbers that belong to mixture $f(x)$

```
seed the random number generator (010101)
Initialize array $d$ of the size of the random numbers generated

For i in 1..n LOOP
    $\zeta$ = generate a r.v. from $U[0, 1]$
    IF $\zeta<\gamma$ THEN
        d[i]= generate a random number from $\phi_0 \sim N(\sigma_0, \mu_0)$
    ELSE
        d[i]= generate a random number from $\phi_1 \sim N(\sigma_1, \mu_1)$
    ENDIF
END LOOP

output d which will now contain n random variables drawn from the above probability density function.
```

Function to implement the mixture random variable algorithm

```
In[190]:= Remove["Global`*"];
          gDebug = False;
```

```
In[192]:= (*this function below imlements the above algorithm*)
          processPartTwo[γ_, μ0_, σ0_, μ1_, σ1_, n_] := Module[{d = {}, k, ξ},
            ξ = Table[RandomReal[], {k, 1, n}];
            d = Table[ If[ξ⟦k⟧ < γ, RandomReal[NormalDistribution[μ0, σ0]],
                RandomReal[NormalDistribution[μ1, σ1]]], {k, 1, n}]
           ]
```

Now generate 2 sets of numbers each 10000 long, one for γ = .75 and the second for γ = .25

```
In[193]:= μ0 = 0; σ0 = 1; μ1 = 3; σ1 = 1; n = 10 000;
          SeedRandom[010 101];
          γ = .75;  setA = processPartTwo[γ, μ0, σ0, μ1, σ1, n];
          γ = .25;  setB = processPartTwo[γ, μ0, σ0, μ1, σ1, n];
```

Part 2 (b)

Generate 2 plots, one for γ = .75 and one for γ=.25 for number of random variables=10000 generated in part (a) overlaid by histogram.
First define the mixture density function (the true density). Please see appendix for the code that overlays the histogram and the mixture function called postprocessPartTwo[]. Moved below to the appendix to reduce code clutter in the main report.

```
In[197]:= mixtureDensity[x_, γ_, μ0_, σ0_, μ1_, σ1_] :=
            γ PDF[NormalDistribution[μ0, σ0], x] + (1 - γ) PDF[NormalDistribution[μ1, σ1], x];
```

Now call the above on the 2 sets of 10000 numbers generated in part (a) and display the result

This function makes a histogram which is scaled to be used to overlay density plots, or other functions.
Input: originalData: this is an array of numbers which represents the data to bin

nBins: number of bins

output: the histogram itself but scaled such that area is ONE

In[198]:=

```
Needs["BarCharts`"]
nmaMakeDensityHistogram[originalData_, nBins_] :=
 Module[{freq, binSize, from, to, scaleFactor, j, a, currentArea},
  to = Max[originalData];
  from = Min[originalData];
  binSize = (to - from) / nBins;
  freq = BinCounts[originalData, binSize];
  currentArea = Sum[binSize * freq[[i]], {i, nBins}];
              freq
  freq = ------------;
         currentArea
  a = from;
  Table[{a + (j - 1) * binSize, freq[[j]], binSize}, {j, 1, nBins}]
 ]
```

In[200]:=

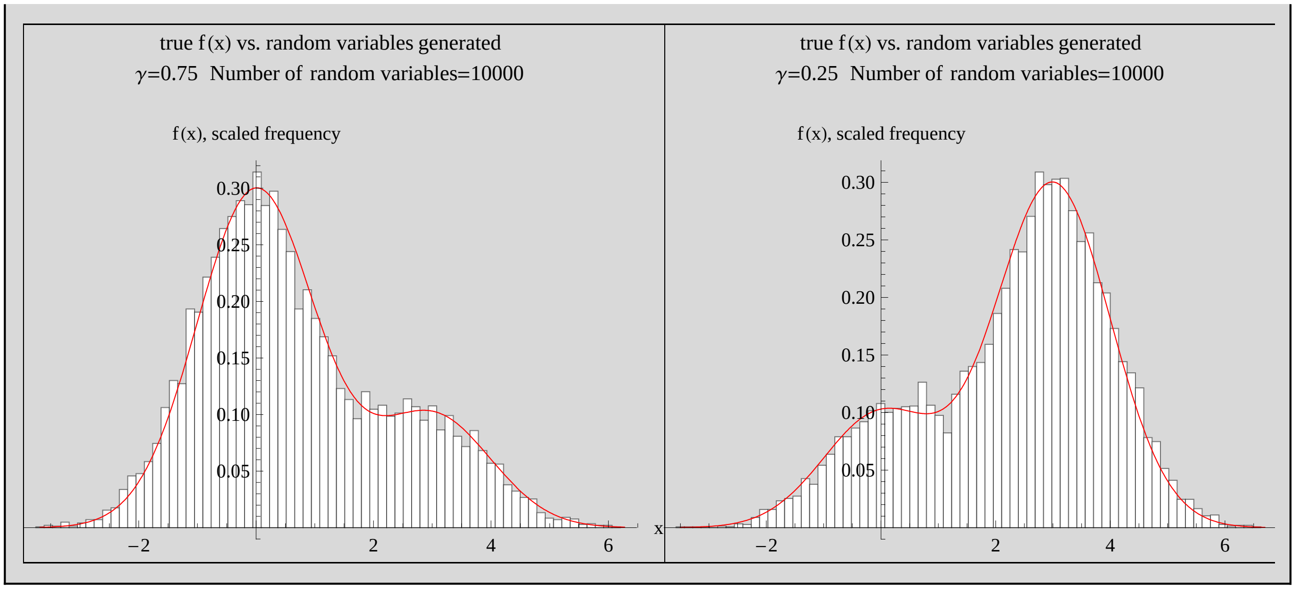
```
postprocessPartTwo[d_, γ_, μ0_, σ0_, μ1_, σ1_, nRandomVariables_, imageSize_] :=
 Module[{freq, p, pList, xFrom, xTo, scaleFactor, maxSampled, sampled,
   gz = {}, fz0, fz1, x, maxBin, imSize = imageSize, nBins = 70, from},
  xFrom = Min[d];
  xTo = Max[d];
  gz = nmaMakeDensityHistogram[d, nBins];
  pList = GeneralizedBarChart[gz, BarStyle → White, ImageSize → imSize];

  p = Plot[mixtureDensity[x, γ, μ0, σ0, μ1, σ1], {x, xFrom, xTo},
    AxesOrigin → {0, 0}, PlotRange → All, PlotLabel → "Analytical plot of f(x)",
    ImageSize → imSize, (*PlotStyle→{Dashed,Red}*)PlotStyle → {Red}];

  Show[{pList, p},
   PlotLabel → Style["true f(x) vs. random variables generated\n" <> "γ=" <> ToString[γ] <>
      "  Number of random variables=" <> ToString[nRandomVariables] <> "\n", 10],
   AxesLabel → {"x", "f(x), scaled frequency"}]
 ]
```

```
In[201]:= γ = .75;  p1 = postprocessPartTwo[setA, γ, μ0, σ0, μ1, σ1, n, 300];
          γ = .25;  p2 = postprocessPartTwo[setB, γ, μ0, σ0, μ1, σ1, n, 300];
          Grid[{{p1, p2}}, Frame → All, Spacings → 0, ItemSize → Full]
```

Out[203]=



Comment and analysis on result of part 2 (b) plots

In the left plot, γ=.75, hence 75% of the mixture comes from ϕ_0 which has a mean of 0, hence we would expect that at zero the bulk of the concentration, which what the plots shows to be the case (since both concentration have the same variance). Hence there should be more random numbers generated from this mixture around x=0 as well, and we see from the histogram that this is the case.
In the right plot, now γ=.25, hence 75% of the concentration will come the ϕ_1 distribution which has mean of 3. Hence again, we see that more random numbers are generated around 3 than anywhere else. These plots also show that the random numbers generated will have a probability density which will converge to the $f(x)$ given as more and more random variables are generated.

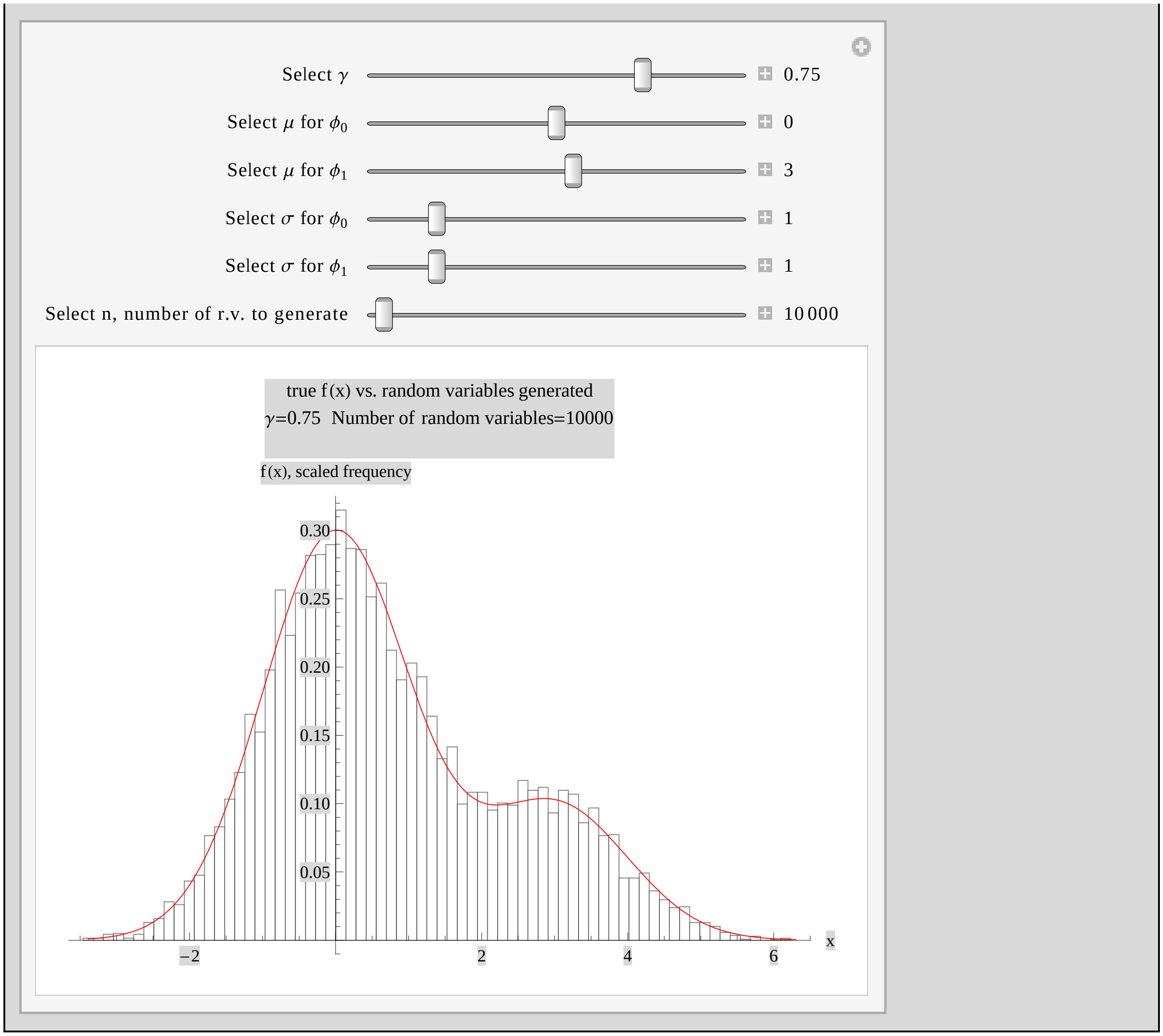
Simulation program for problem 2

GUI to simulate part 2(b) of the project

In[204]:=

```
m = Manipulate[
  postprocessPartTwo[processPartTwo[γ, μ0, σ0, μ1, σ1, n], γ, μ0, σ0, μ1, σ1, n, 400],
  {{γ, .75, "Select γ"}, 0, 1, .1, Appearance → "Labeled"}
  , {{μ0, 0, "Select μ for ϕ0"}, -30, 30, .1, Appearance → "Labeled"}
  , {{μ1, 3, "Select μ for ϕ1"}, -30, 30, .1, Appearance → "Labeled"}
  , {{σ0, 1, "Select σ for ϕ0"}, 0.1, 6, .1, Appearance → "Labeled"}
  , {{σ1, 1, "Select σ for ϕ1"}, 0.1, 6, .1, Appearance → "Labeled"}
  , {{n, 10 000, "Select n, number of r.v. to generate"}, 10 000, 100 000, 10 000,
   Appearance → "Labeled"}, AutorunSequencing → {{1, 15}, {2, 25}, {3, 25}, {4, 25}, {5, 25}}
 ]
```

Out[204]=



Select γ 0.75

Select μ for ϕ_0 0

Select μ for ϕ_1 3

Select σ for ϕ_0 1

Select σ for ϕ_1 1

Select n, number of r.v. to generate 10 000

true f(x) vs. random variables generated

γ=0.75 Number of random variables=10000

In[205]:=

```
(*Export["m.swf",m,"RepeatAnimation"→True,"CompressionMethod"->None]*)
```