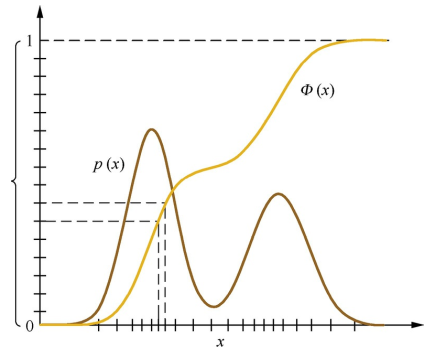
# 重要性采样的理解

如果我们要计算一个函数在x符合分布下的期望，我们可以如下图中这样，计算出累计分布函数，然后在纵轴上均匀采样，然后映射到横轴上，得到**符合分布的样本**，将这些样本计算再求和即可。



但是，有时候累积分布函数不易求得，我们可以选择一个容易计算累计分布函数的分布，那么，以上期望我们可以采用以下公式求得：

，其中

是采用以上采样方法从的累积分布函数中采样得到的符合分布的样本，是两个分布在该样本点上的概率密度比值（权重）。

示例代码如下，假设真实分布（这里是均值为1，方差为1的高斯分布）是难以计算累计分布函数是难以计算的，我们要计算函数在该分布下的期望，我们分布取为均值为1.0，方差为1.0，0.5，2.0的分布来进行重要性采样。由于python本身有按照高斯分布进行采样的函数，因此我们这里没有再实现采样的累积分布函数实现均匀分布到高斯分布的转换，即按照高斯分布进行采样（上图中的方法），而是直接使用np**.**random**.**normal函数。

从结果看出，虽然我们使用了不同的，但是我们计算出来的期望是相近的。

|  |
| --- |
| **import** numpy **as** np  **import** math  **import** matplotlib**.**pyplot **as** plt  **def** f**(**x**):**  **return** x **\*** x  **def** gaussian**(**x**,**u**,**sigma**):**  """  param x:要计算概率密度值的点  param u:均值  param sigma:方差  return x的概率密度值  """  **return** math**.**exp**(-(**x**-**u**)\*\***2**/(**2**\***sigma**\***sigma**))/**math**.**sqrt**(**2**\***math**.**pi**\***sigma**\***sigma**)**  **def** importance\_sampling\_test**(**ori\_sigma**,**sample\_sigma**):**  """  param ori\_sigma:原始分布p(x)的方差  param sample\_sigma:采样分布p~(x)的方差  return  """  origin **=** **[]**  **for** n **in** range**(**10**):**  #进行10次计算  Sum **=** 0  **for** i **in** range**(**100000**):**  a **=** np**.**random**.**normal**(**1.0**,**ori\_sigma**)**  Sum **+=** f**(**a**)**  origin**.**append**(**Sum**)**    isample **=** **[]**  **for** n **in** range**(**10**):**  Sum2 **=** 0  **for** i **in** range**(**100000**):**  a **=** np**.**random**.**normal**(**1.0**,**sample\_sigma**)** #从正太分布采样出来的x  ua = gaussian(a,1.0,sample\_sigma) #计算采样概率密度  na = gaussian(a,1.0,ori\_sigma) #计算原始概率密度  Sum2 += f(a)\*na/ua  isample.append(Sum2)    origin = np.array(origin)  isample = np.array(isample)    print(np.mean(origin),np.std(origin))  print(np.mean(isample),np.std(isample))    importance\_sampling\_test(1.0,1.0)  importance\_sampling\_test(1.0,0.5)  importance\_sampling\_test(1.0,2.0)  xs = np.linspace(-5,6,301)  y1 = [gaussian(x,1.0,1.0) for x in xs]  y2 = [gaussian(x,1.0,0.5) for x in xs]  y3 = [gaussian(x,1.0,2.0) for x in xs]  fig = plt.figure(figsize=(8,5))  plt.plot(xs,y1,label="sigma=1.0")  plt.plot(xs,y2,label="sigma=0.5")  plt.plot(xs,y3,label="sigma=2.0")  plt.legend()  plt.show() |

