Chi-Squared Test Pseudocode

This version of the document is dated 2019-02-06.

Introduction

This short article sets forth pseudocode for performing a [chi-squared test](#) using observed frequencies and probabilities.

In the pseudocode below—

- the **pseudocode conventions** apply,
- `ALMOSTZERO` is a number very close to 0 (for example, 10^{-100}), and
- `EPSILON` is a number setting an estimation tolerance for the p-value calculation (for example, 10^{-14}).

The following are the parameters for `ChiSquaredTest` below:

- `probabilities` is a list of probabilities for each event (ranging from 0, never, to 1, always).
- `counts` is a list with the same size as `probabilities` and gives, for each event, the number of times that event occurs. The sum of all items in `counts` should be at least 5.

`ChiSquaredTest` returns a list containing two items: the first item is the chi-squared statistic, and the second item is the *p-value* (the probability that a random sample's chi-squared statistic would be greater than or equal to that of the given sample). (Note that if `probabilities` has only one item, the p-value will be meaningless.)

Pseudocode

```
METHOD DoubleFactorial(x)
  if x<=1: return 1
  ret=1
  x=floor(x)
  smod = rem(x,2)
  k=0
  if smod==0: k=floor(x/2)
  if smod==1: k=floor((x-1)/2)+1
  i=0
  while i<k
    ret=ret*2*(i+1)-smod
    i=i+1
  end
  return ret
END METHOD

METHOD GammaOfHalfInteger(x)
  if x==0: return infinity
  if x<0: return error
  // Computes gamma(x/2) assuming `x` is an integer
  return DoubleFactorial(x-2)*sqrt(pi)/pow(2,(x-1)*0.5)
END METHOD

METHOD GammaRegQ(a,b)
  // Computes the Q version of the regularized gamma
  // function for `a` divisible by 1/2
```

```
    if a==0: return infinity
    if a<0: return error
    mul=pow(b,a)*exp(-b)
    // NOTE: All uses of this function assume `a` is divisible
    // by 1/2.  For general `a`, replace `GammaOfHalfInteger(a * 2)`
    // with a call to the full-fledged gamma function like
    // `gamma(a)`.
    mul = mul / GammaOfHalfInteger(a * 2)
    // NOTE: Continued fraction calculation technique
    // described in Thompson and Barnett, "Coulomb and
    // Bessel functions of complex arguments and order",
    // June 1986.
    ret=ALMOSTZERO // For example, 10^-100
    dLast=0
    cLast=ret
    i=0
    while i < 100
      // Get next convergent of continued fraction
      if i==0
        num=1.0
        den=1.0-a+b
        dencount=3.0
        numcount=1.0
      else
        num=numcount*(a-numcount)
        den=dencount-a+b
        dencount+=2.0
        numcount+=1.0
      end
      // Calculate more of continued fraction
      c=den+num/cLast
      d=den+num*dLast
      if d==0: d=ALMOSTZERO
      if c==0: c=ALMOSTZERO
      d=1.0/d
      delta=d*c
      ret*=delta
       // For example 10^-14
      if abs(delta-1.0)<EPSILON: break
      dLast=d
      cLast=c
      i=i+1
    end
    return mul*ret
END METHOD

METHOD ChiSquaredTest(probabilities, counts)
  if size(probabilities)!=size(counts): return error
  // Calculate number of samples
  numSamples=0
  i=0
  while i<size(counts)
    numSamples=numSamples+counts[i]
    i=i+1
  end
  // Calculate chi-squared statistic
  chisq=0
  i=0
  while i<size(probabilities)
    ex=probabilities[i]*numSamples*1.0
    chisq=chisq+pow(counts[i]-ex,2)/ex
    i=i+1
```

```
        end
    // Calculate p-value
    rga=(size(probabilities)-1)*0.5
    pvalue=GammaRegQ(rga,chisq*0.5)
    // Return statistic and p-value as a list
    return [chisq, pvalue]
END METHOD
```