Steekproefalgoritmes in Excel

In Excel kun je in de module Visual Basic eigen algoritmes schrijven die als User Defined Functions worden aangeduid (UDF). Een UDF is in feite een Excelfunctie die niet standaard in Excel is opgenomen, maar door jou zelf is geschreven. We kunnen deze toepassing inzetten om een functie te schrijven die steekproefgroottes voor ons genereert. Om dit te doen moet je de volgende instructies volgen:

Stap 1: Parameters definiëren

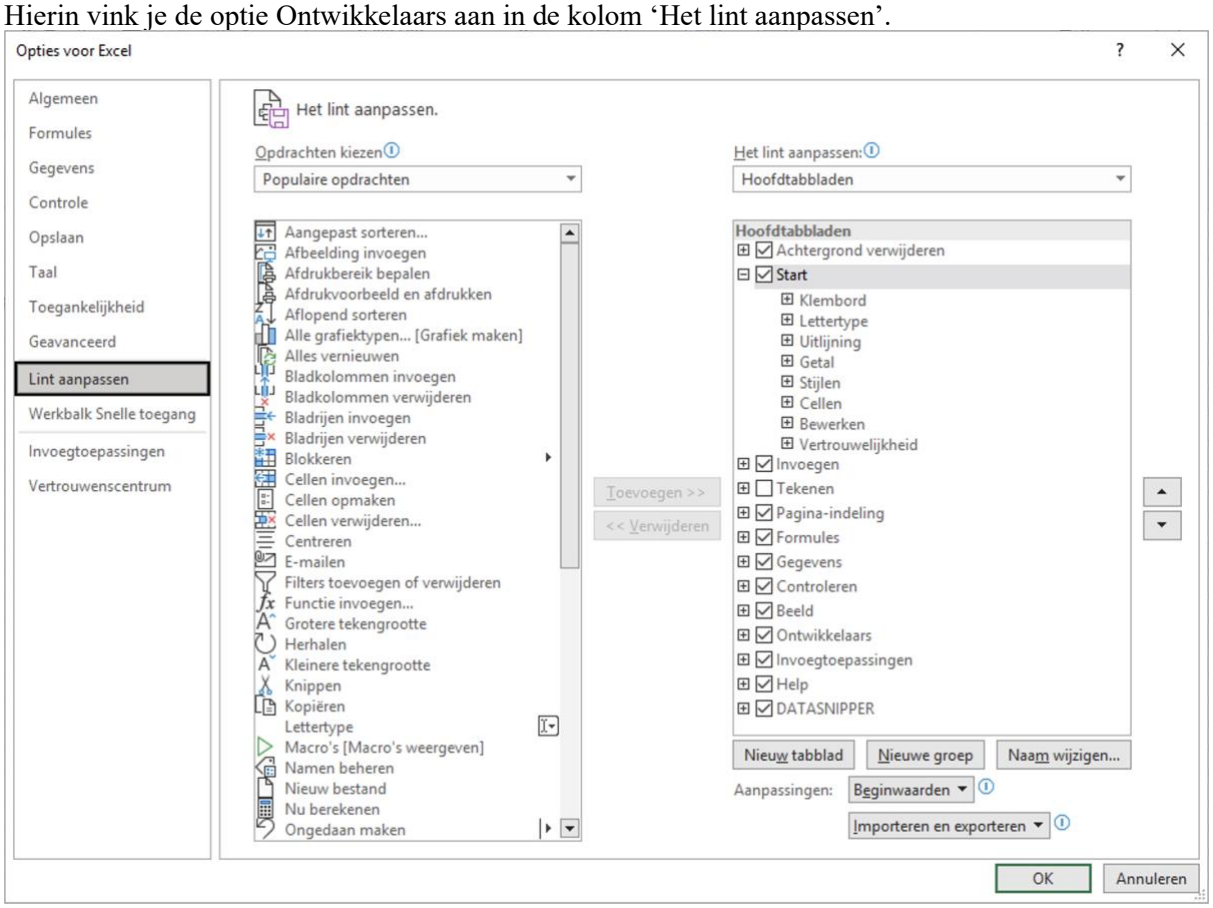
Definieer eerst de benodigde gegevens voor het berekenen van de steekproefomvang in Excel. Plaats in een aantal cellen de volgende gegevens:

- Populatieomvang (N)
- Uitvoeringsmaterialiteit (p)
- Verwachte fout in percentages (k)
- Gewenste betrouwbaarheid (1-β)

	A	B	C
1			
2			
3		Populatieomvang (N)	1.000.000
4		Uitvoeringsmaterialiteit (p)	50.000
5		Verwachte fout in % (k)	0%
6		Betrouwbaarheid (1-β)	95%
7			

Stap 2: Het ontwikkelaars lint activeren

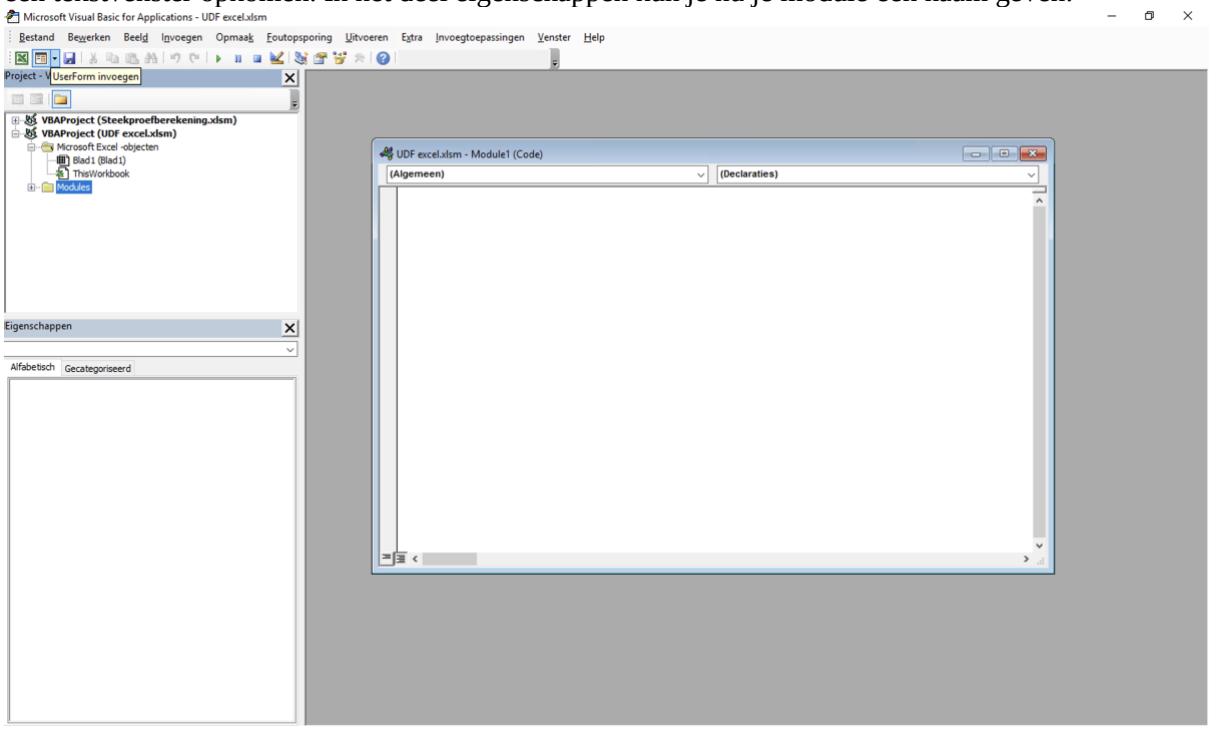
Open Excel en druk op bestand/Opties. In het venster dat opkomt navigeer je naar Lint aanpassen. Hierin vink je de optie Ontwikkelaars aan in de kolom 'Het lint aanpassen'.



Het lint ziet er nu als volgt uit:

Stap 3: VBA openen en een module openen

Vervolgens navigeer je naar Ontwikkelaars in het lint en druk je op Visual Basic. Druk op de rechtermuiskop bij VBAProject (jouw Excelnaam). Selecteer vervolgens Invoeren/Module. Er zal nu een tekstvenster opkomen. In het deel eigenschappen kun je nu je module een naam geven.



Stap 4: De formule aanmaken.

In dit venster kun je een algoritme invoeren in VBA. Je kunt deze helemaal zelf schrijven, maar ook onderstaande code kopiëren. De code in deze column voor de Poisson en Binomiale verdeling zijn een verdere uitwerking van het werk van Stewart (2012). Mocht je de werking verder willen onderzoeken raadpleeg dan deze bron. Het algoritme voor de hypergeometrische verdeling kent dezelfde opbouw, alleen dan gebruikmakend van een andere verdeling. In de onderstaande code staat in het groen aangegeven wat er precies gebeurt in het algoritme. Het algoritme wordt door Excel VBA stapsgewijs regel per regel uitgevoerd.

Poisson

```
'Start met het definieren van de functie en welke variabelen je de gebruiker wilt vragen'
Function PoissonSample(Populatie As Double, Materialiteit As Double, FoutVerwacht As Double, Betrouwbaarheid As
Double)
   'Definieer extra variabelen om mee te kunnen werken'
   Dim Risk As Double
   Dim pE As Double
   Dim pT As Double
   Dim N As Double
   'Bereken de benodigde variabelen op basis van de gegeven variabelen'
   Risk = 1 - Betrouwbaarheid
   pT = Materialiteit / Populatie
   pE = FoutVerwacht
   'Geef een error wanneer verkeerde waardes zijn ingevoerd'
   If Risk <= 0 Or Risk >= 1 Or pE < 0 Or pE >= 1 Or pT <= 0 Or pT >= 1 Then
      PoissonSample = CVErr(xlErrNum)
   'Het algoritme'
    'Merk op dat k wordt afgerond naar boven. Dit is nodig omdat er slechts spaken van zijn van gehele fouten.'
   Else
      N = 1
      k = Application.WorksheetFunction.RoundUp(pE * N, 0)
      While Application.WorksheetFunction.GammaDist(N * pT, 1 + k, 1, True) < Betrouwbaarheid And N <= 20000
         N = N + 1
         k = Application.WorksheetFunction.RoundUp(pE * N, 0)
      Wend
      'sluit het algoritme af, met een als functie die checkt of de uitkomst inderdaad een uitkomst geeft met een goede
betrouwbaarheid, anders een error'
      PoissonSample = IIf(Application.WorksheetFunction.GammaDist(N * pT, 1 + k, 1, True) >= 1 - Risk, N,
CVErr(xlErrNA))
   End If
End Function
```

Binomiaal

```
'Start met het definieren van de functie en welke variabelen je de gebruiker wilt vragen'
Function BinomSample(Populatie As Double, Materialiteit As Double, FoutVerwacht As Double, Betrouwbaarheid As
Double)
   'Definieer extra variabelen om mee te kunnen werken'
   Dim Risk As Double
   Dim pE As Double
   Dim pT As Double
   Dim N As Double
   Dim k As Double
   'Bereken de benodigde variabelen op basis van de gegeven variabelen'
   Risk = 1 - Betrouwbaarheid
   pT = Materialiteit / Populatie
   pE = FoutVerwacht
   'Geef een error wanneer verkeerde waardes zijn ingevoerd'
   If Risk <= 0 Or Risk >= 1 Or pE < 0 Or pE >= 1 Or pT <= 0 Or pT >= 1 Then
      BinomSample = CVErr(xlErrNum)
   'Het algoritme'
   ‘Merk op dat k wordt afgerond naar boven. Dit is nodig omdat er slechts spaken van zijn van gehele fouten.'
   Else
      N = Application.WorksheetFunction.RoundUp(Log(Risk) / Log(1 - pT), 0)
      k = Application.WorksheetFunction.RoundUp(pE * N, 0)
      While Application.WorksheetFunction.BinomDist(k, N, pT, True) > Risk And N <= 20000
         N = N + 1
         k = Application.WorksheetFunction.RoundUp(pE * N, 0)
      Wend
       'sluit het algoritme af, met een als functie die checkt of de uitkomst inderdaad een uitkomst geeft met een goede
betrouwbaarheid, anders een error'
      BinomSample = IIf(Application.WorksheetFunction.BinomDist(k, N, pT, True) <= Risk, N, CVErr(xlErrNA))
   End If
End Function
```

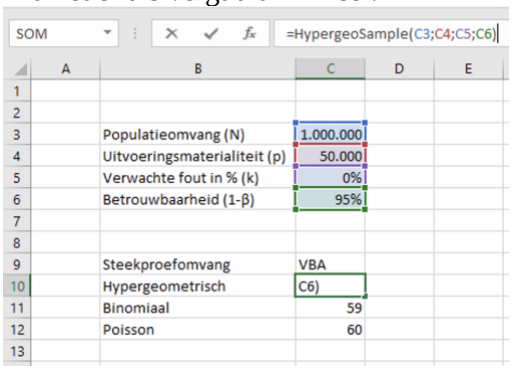
Hypergeometrisch

```
'Start met het definieren van de functie en welke variabelen je de gebruiker wilt vragen'
Function HypergeoSample(Populatie As Double, Materialiteit As Double, FoutVerwacht As Double, Betrouwbaarheid
As Double)
   'Definieer extra variabelen om mee te kunnen werken'
   Dim N As Double, k As Double, L As Double
   'Bereken de benodigde variabelen op basis van de gegeven variabelen'
   Risk = 1 - Betrouwbaarheid
   pT = Materialiteit / Populatie
   pE = FoutVerwacht
   p = Populatie
   'Geef een error wanneer verkeerde waardes zijn ingevoerd'
   If Risk <= 0 Or Risk >= 1 Or pE < 0 Or pE >= 1 Or pT <= 0 Or pT >= 1 Then
      HypergeoSample = CVErr(xlErrNum)
   Else
   'Het algoritme'
   'Merk op dat k en L worden afgerond naar boven. Dit is nodig omdat er slechts spaken van zijn van gehele fouten.'
      N = Application.WorksheetFunction.RoundUp(1, 0)
      k = Application.WorksheetFunction.RoundUp(pE * N, 0)
      L = Application.WorksheetFunction.RoundUp(pT * p, 0)
      While Application.WorksheetFunction.HypGeom_Dist(k, N, L, p, True) > Risk And N <= 20000
         N = N + 1
         k = Application.WorksheetFunction.RoundUp(pE * N, 0)
         L = Application.WorksheetFunction.RoundUp(pT * p, 0)
      Wend
       'sluit het algoritme af, met een als functie die checkt of de uitkomst inderdaad een uitkomst geeft met een goede
betrouwbaarheid, anders een error'
      HypergeoSample = IIf(Application.WorksheetFunction.HypGeom_Dist(k, N, L, p, True) <= Risk, N,
CVErr(xlErrNA))
   End If
End Function
```

Stap 5: De formule invullen

Vul vervolgens de gegevens die je hebt gedefinieerd in stap 1 in de formule. Er rolt nu een bruikbare minimale steekproefomvang uit. Het voorbeeld dat we eerder handmatig hadden uitgerekend pakt via dit algoritme als volgt uit:
Populatieomvang (N) = 1.000.000
Materialiteit (p) = 5% = 50.000
Verwachte foutpercentage = 0%
Betrouwbaarheid (1-β) = 95%
=HypergeoSample(1000000;50.000;0;95%) = 59
=BinomSample(1.000.000;50.000;0;95%) = 59
=PoissonSample(1.000.000;50.000;0;95%) = 60

Dit ziet er als volgt uit in Excel:



In het bijgevoegde Excelrekenblad zijn de UDFs al opgenomen waarmee je kunt controleren of jouw eigen rekenbladen of auditsoftware de juiste steekproefomvang berekent en welke parameters daarvoor ingevoerd moeten worden.