Bayer CropScience

Nutzung der SAS-Funktionalität aus MS-Office Applikationen

- Das SAS-System bietet Lösungen für komplexe analytische Probleme. Die Komplexität des Programms selbst bringt jedoch oft für den Anwender die Notwendigkeit, in einer ihm nicht vertrauten Entwicklungsumgebung zu agieren. Ab der SAS-Version 8 besteht die Möglichkeit der Steuerung des Systems aus einem anderen Programm heraus über OLE.
- In der vorgestellten Anwendung erfolgt die Datenerfassung in MS Excel. Die Datenanalyse erfolgt über das im Hintergrund laufende SAS-System. Die Analyseergebnisse werden an MS Excel zurückgegeben bzw. in Form von RTF-Dateien gespeichert. Ein Vorteil der vorgestellten Lösung besteht auch darin, dass die Auswerteroutinen des SAS-Systems bereits als validiert gelten.

Armin Schüler
D-AD / Data Management
Bayer CropScience GmbH
Industriepark Höchst
Building K 607
D-65926 Frankfurt am Main
Tel. +49 (69) 305 – 24864
armin.schueler@bayercropscience.com

Dr. Harald Schulz
Development
SLAT IT-Consulting GmbH
Auf der Lind 9
D-65529 Waldems
Tel. +49 (6126) 99 75 0
Harald.Schulz@SLAT.de

Inhalt

1. Vorgehen bei der Entwicklung
2. Zugriff auf SAS
3. Datenübergabe von SAS an Excel
4. Zugriff auf MS Word
5. Fehlersuche
6. Stolpersteine
7. Beispiel
8. Programmstruktur
9. Aufbau der Ablage
10. LD50-Problematik
11. Erstellen der Dokumentation
12. Zusammenfassung

Vorgehen bei der Entwicklung

Vorgehen bei der Entwicklung

- Vorhanden
  - SAS-Anwendung LD50-Berechnung für einen Organismus
  - Verschiedene Excel-Eingabesheets für verschiedene Organismen
  - Daten werden vom Excel-Sheet in das SAS abgetippt

Vorgehen bei der Entwicklung

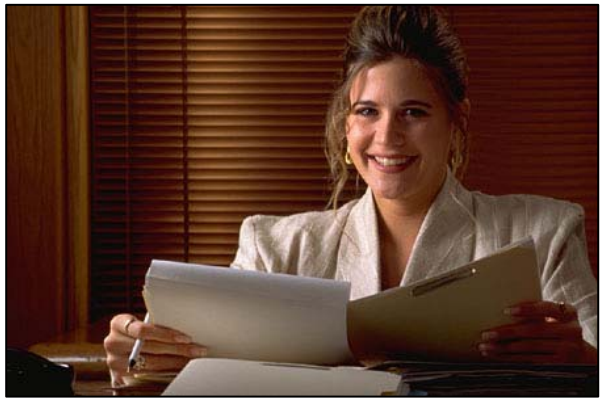
- User-Anforderung:
  - die SAS-Berechnung muss vom Handling vereinfacht werden
  - Vorhandene Excel-Eingabe-Tabellen sollen nicht verändert werden
  - gegen GLP validiert

Vorgehen bei der Entwicklung

- Ziel:
  - benutzerfreundliche Lösung
    (User arbeitet im MS Office-Umfeld)
  - neue Guidelines einarbeiten

- Lösung:

  - Steuerung des SAS über Excel



Vorgehen bei der Entwicklung

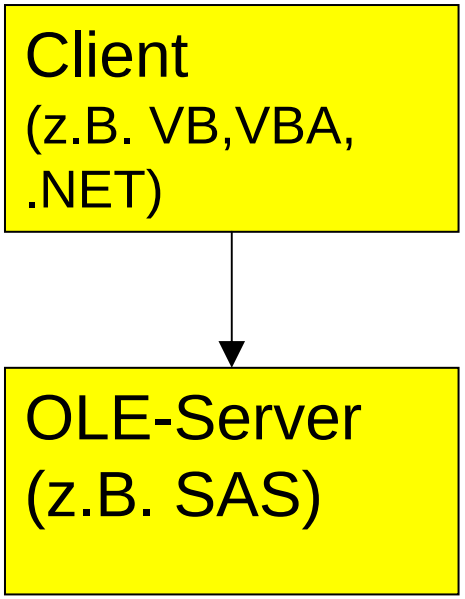
- Steuerung der SAS-Anwendung von Excel (VBA)
- Auflösung der SAS-Module → Excel-Unterprogramm
- Erweiterung des SAS/Excel-Programms
- Einbinden von Word zur automatisierten Dokumentationserstellung

Zugriff auf SAS

Zugriff auf SAS

- SAS unterstützt
  - DDE
  - OLE 1.0 seit SAS Release 6.08
  - OLE 2.0 seit SAS Version 8
    → OLE-Automation

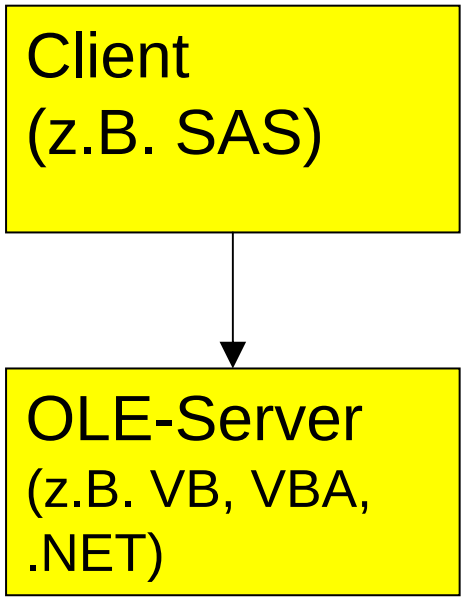
OLE-Automation



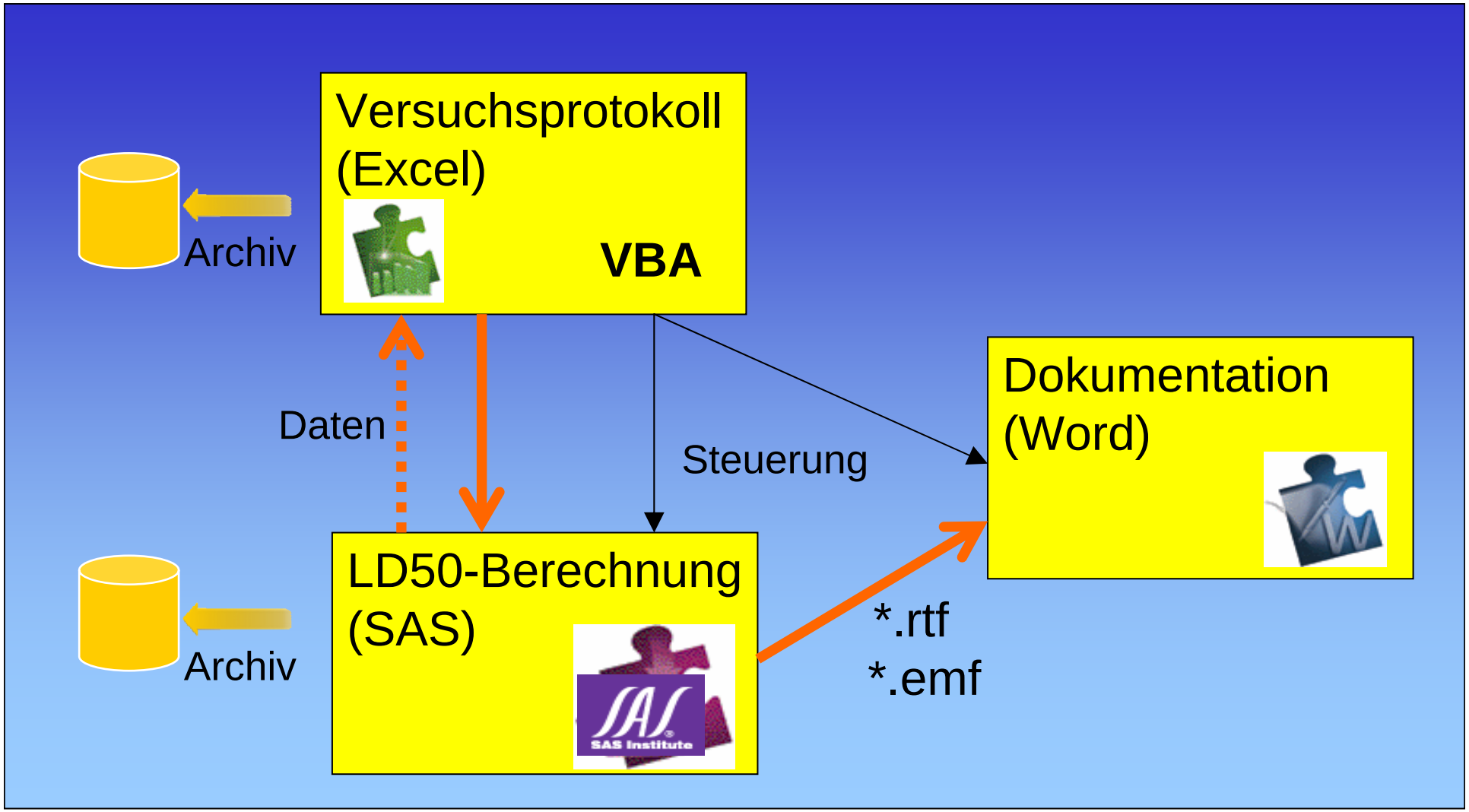
- SAS als OLE-Server
- SAS wirkt wie skriptgesteuert

OLE-Automation

- Alternativ Excel als OLE-Server



Beispiel LD50



Zugriff auf SAS von Excel

- Beispiele in der Beschreibung sind stark vereinfacht

```
'Zugriff auf SAS
   i = SetStatus("Zugriff auf SAS", 6)
   On Error Resume Next

   ' Try first to use an existing instance.
   blnWasntRunning = False
   Set objSAS = GetObject(, "SAS.Application.8")
   If Err Then
       ' Oops! SAS isn't running, so start it.
       blnWasntRunning = True
       Set objSAS = CreateObject("SAS.Application.8")
   End If
   objSAS.Wait = True
   If objSAS Is Nothing Then
       MsgBox "Error " & ": " & "SAS läuft nicht" + _
              Chr(13) & "bitte starten Sie Ihren Computer neu", _
              vbCritical, Application.ActiveWorkbook.Name + " CallSAS"
       i = ResetStatus()
       Application.ScreenUpdating = True
       Exit Sub
   End If

   On Error GoTo Err_Handler
```

```
'Zugriff auf SAS
   i = SetStatus("Zugriff auf SAS", 6)
   On Error Resume Next

   ' Try first to use an existing instance.
   Set objSAS = GetObject(, "SAS.Application.8")
   If Err Then
       ' Oops! SAS isn't running, so start it.
       blnWasntRunning = True
       Set objSAS = CreateObject("SAS.Application.8")
   End If
   objSAS.Wait = True
   If objSAS Is Nothing Then
       MsgBox "Error " & ": " & "SAS läuft nicht" + _
              Chr(13) & "bitte starten Sie Ihren Computer neu", _
              vbCritical, Application.ActiveWorkbook.Name + " CallSAS"
       i = ResetStatus()
       Application.ScreenUpdating = True
       Exit Sub
   End If

   On Error GoTo Err_Handler
```

```
'Zugriff auf SAS
    i = SetStatus("Zugriff auf SAS", 6)

    On Error Resume Next

    Set objSAS = GetObject(, "SAS.Application.8")
    If Err Then
        blnWasntRunning = True
        Set objSAS = CreateObject("SAS.Application.8")
    End If
    objSAS.Wait = True
    If objSAS Is Nothing Then
        MsgBox "Error " & ": " & "SAS läuft nicht" + _
               Chr(13) & "bitte starten Sie Ihren Computer neu", _
               vbCritical, Application.ActiveWorkbook.Name + " CallSAS"
        i = ResetStatus()
        Application.ScreenUpdating = True
        Exit Sub
    End If

    On Error GoTo Err_Handler
```

```
'Zugriff auf SAS
   i = SetStatus("Zugriff auf SAS", 6)

   On Error Resume Next

   Set objSAS = GetObject(, "SAS.Application.8")
   Set objSAS = GetObject(, "SAS.Application.8")

   If Err Then
      blnWasntRunning = True

      Set objSAS = CreateObject("SAS.Application.8")
   End If
   objSAS.Wait = True

   objSAS Is Nothing
      MsgBox "Error " & ": " & "SAS läuft nicht" + _
             Chr(13) & "bitte starten Sie Ihren Computer neu"

      →“Bitte starten Sie Ihren Computer neu“
      i = ResetStatus()
      Application.ScreenUpdating = True
      Exit Sub
   End If

   On Error GoTo Err_Handler
```

```
'Zugriff auf SAS
   i = SetStatus("Zugriff auf SAS", 6)
   On Error Resume Next

   ' Try first to use an existing instance.
   Set objSAS = GetObject(, "SAS.Application.8")
   If Err Then
       ' Oops! SAS isn't running, so start it.
       blnWasntRunning = True
       Set objSAS = CreateObject("SAS.Application.8")
   End If
   objSAS.Wait = True
   If objSAS Is Nothing Then
       MsgBox "Error " & ": " & "SAS läuft nicht" + _
              Chr(13) & "bitte starten Sie Ihren Computer neu", _
              vbCritical, Application.ActiveWorkbook.Name + " CallSAS"
       i = ResetStatus()
       Application.ScreenUpdating = True
       Exit Sub
   End If

   On Error GoTo Err_Handler
```

Zugriff auf SAS beenden

```
'Only shut down SAS if it wasn't running before.
   If blnWasntRunning Then
       objSAS.Quit
   Else
       objSAS.Visible = True
   End If

   Set objSAS = Nothing
```

Datenübergabe SAS → Excel

Datenübergabe SAS → Excel

- Feedback von SAS an Excel als
  - ResultsString → Text

    oder
  - RC → Zahl

```
'Übergabe der LD-50-Werte an Excel
       .Submit ("data _null_;")
       .Submit ("   set abbott_out;")
       If intVersuchsart < 30 Then
           .Submit ("   where trt ='" & strTRT & "';")
       End If
       .Submit ("   if nmiss(value_c) then ld50_value='n.d.       n.d.       n.d.';")
       .Submit ("   else ld50_value = input(value_c,$8.) || input(lowercl_c,$8.)|| _
                   input(uppercl_c,$8.);")
       .Submit ("run;")

       dtePause = Now()
       Do
           strLD50 = “0”
           strCI_Lower = “0”
           strCI_Upper = “0”
           If Now() - dtePause > 0.0001 Then Exit Do
           DoEvents
       Loop While.busy = True Or .resultstring = strOldResultString

       If .resultstring <> strOldResultString Then
           strLD50 = Trim(Left(.resultstring, 8))
           strCI_Lower = Trim(Mid(.resultstring, 9, 8))
           strCI_Upper = Trim(Mid(.resultstring, 17, 8))
           strOldResultString = .resultstring
       End If
```

```
'Übergabe der LD-50-Werte an Excel
       .Submit ("data _null_;")
```

```
.Submit ("ld50_value =
    input(value_c,$8.) || _
    input(lowercl_c,$8.)|| _
    input(uppercl_c,$8.);")
```

```
    dtePause = Now()
    Do
        strLD50 = “0”
        strCI_Lower = “0”
        strCI_Upper = “0”
        If Now() - dtePause > 0.0001 Then Exit Do
        DoEvents
    Loop While.busy = True Or .resultstring = strOldResultString
```

```
strLD50 = Left(.resultstring, 8)
strCI_Lower = Mid(.resultstring, 9, 8)
strCI_Upper = Mid(.resultstring, 17, 8)
```

```
'Übergabe der LD-50-Werte an Excel
       .Submit ("data _null_;")
       .Submit ("ld50_value =
           input(value_c,$8.) || _
           input(lowercl_c,$8.)|| _
           input(uppercl_c,$8.);")
    dtePause = Now()
    Do
        ...
    Loop While ...
        DoEvents
    Loop While.busy = True Or .resultstring = strOldResultString

    strLD50 = Left(.resultstring, 8)
    strCI_Lower = Mid(.resultstring, 9, 8)
    strCI_Upper = Mid(.resultstring, 17, 8)
```

Zugriff auf MS Word

Zugriff auf MS Word

- Text
  rtf-Format
- Grafik
  EMF-Format

Fehlersuche

Fehlersuche

- SAS visible (über Flag in Excel-VBA)
  Im Schrittbetrieb Steuerung von SAS

- Debug in SAS
  - Debug inplicit
  - Symbolgen
    →Erstellen einer SAS-log-Datei

Debug-Hilfen aktivieren

```
objSAS.Visible = blnSASVisible ‘true oder false

    'Debug-Hilfe
    If blnDebugVal = True Then ‘SAS-LOG wird erzeugt
        objSAS.Submit ("options mprint;")
        objSAS.Submit ("options symbolgen;")
    End If
```

```
If blnDebugVal = True Then
        objSAS.Submit ("options nomprint;")
        objSAS.Submit ("options nosymbolgen;")
    End If
```

Debug-Hilfen aktivieren

```
objSAS.Visible = blnSASVisible ‘true / false

'Debug-Hilfe
If blnDebugVal = True Then ‘SAS-LOG
     objSAS.Submit ("options mprint;")
     objSAS.Submit ("options symbolgen;")
End If
```

```
If blnDebugVal = True Then
       objSAS.Submit ("options nomprint;")
       objSAS.Submit ("options nosymbolgen;")
   End If
```

Stolpersteine

Results-Bibliothek löschen

```
'Results-Bibliothek löschen
       If intVersuchsart < 30 Then
           .Submit ("dm 'ODSRESULTS;Clear noconfirm _
                          nomenu nobutton' continue;dm _
                          'ODSRESULTS;Clear noconfirm nomenu _
                          nobutton' continue;")
          Application.SendKeys (Chr(13))
           DoEvents
       End If
```

Results-Bibliothek löschen

```
'Results-Bibliothek löschen
       If intVersuchsart < 30 Then
           .Submit ("dm 'ODSRESULTS;Clear noconfirm _
                           nomenu nobutton' continue;dm _
                           'ODSRESULTS;Clear noconfirm nomenu _
           Application.SendKeys (Chr(13))
           DoEvents
       End If
```

- Versionsunterschiede

```
Leere Anweisung
SAS 8.1: submit(“”);
SAS 8.2: submit(“ “);
```

- Steuerbefehle pro submit statement im VBA müssen im SAS einzeln ausführbar sein.

Beispiel

Schlupfwespe

Im Beispiel

- 7 Dosierungen, davon ein toxischer Standard und eine unbehandelte Kontrolle.
- Bestimmung der Mortalität auf Basis von 3*10 Tieren
- Bestimmung der Reproduktionsraten auf Basis von 15 Tieren

Programmstruktur

Programmstruktur

- ' CallSAS
- ' |--> sasRunPgms
- ' |--> sasStyle 'Ausgabestyle festlegen
- ' |--> sasMortality 'Daten Vorbereitung
- ' |--> sasTreatment
- ' |--> sasReproduction
- ' |--> sasAnalysis 'Auswertung+Ausgabe

SASAnalysis - Auswertung

- Mortalität
  - LD50 Bestimmung unter Berücksichtigung der natürlichen Mortalität (proc probit / optc lackfit)
  - Fisher's Exact test adjustiert mit Bonferroni-Holm (proc multest / holm test fisher)

- Vergleich der Reproduktionsraten
  - Varianzanalyse (proc mixed / Dunnett)
  - Überprüfung der Modellannahmen (proc glm / Levene + QQ-plot; proc univariate / Kolmogorov-Smirnov)

SASAnalysis - Ausgabe

- Ausgabe
  - proc template
  - proc report
  - ods rtf

```
.Submit ("ods select all;")
.Submit ("ods rtf file='" & strDirAusgabe &
           "\docu\Result.rtf'  style=styles.newrtf;")
.Submit ("ods listing close;")
.Submit ("** title see compute before _page_ statement';")
.Submit ("title2;")
.Submit ("footnote '" & strStudId & "';")
.Submit ("proc report data=global_out nowd headskip headline ...
....
.Submit ("  compute after _page_;")
.Submit ("    texta1 = 'LD50: " & strLD50 & " " & Application.Worksheets("Versuchsprotokoll").Range("Treatment_Einheit") & ";
          95 %  Confidence Interval: (" & strCI_Lower & " - " & strCI_Upper & ")';")
.Submit ("    line @1 texta1 $120.;")
.Submit ("    line @1 '* Fisher`s Exact test, two-sided, p-values are adjusted according to Bonferroni-Holm';")
.Submit ("    line @1 '# one-way ANOVA,  p-values are adjusted according to Dunnett';")
.Submit ("  endcomp;")
....
.Submit ("  define dose /order f=7.1 '" & Application.Worksheets
            ("Versuchsprotokoll").Range("Treatment_Einheit") & "'
             width=15 center;")
```

```
.Submit ("ods select all;")
.Submit ("ods rtf file='" & strDirAusgabe &
.Submit ("proc report data=global_out nowd headskip headline ...
....
.Submit ("  compute after _page_;")
.Submit ("    texta1 = 'LD50: " & strLD50 & " " & Application.Worksheets("Versuchsprotokoll").Range("Treatment_Einheit") & ";
.Submit ("define dose /order f=7.1    & Application.Worksheets
          ("Versuchsprotokoll").Range("Treatment_Einheit") & "'
           width=15 center;")
```

ods rtf file=' " & strDirAusgabe & "\docu\Result.rtf'
style=styles.newrtf;

texta1 = 'LD50: " & strLD50 & "
" & Application.Worksheets("Versuchsprotokoll").
Range("Treatment_Einheit") & " ..... ';")

Mortality / Reproduction - 48 hours after treatment							
		Mortality [%]			Reproduction [%]		
Treatment	g ai/ha	Uncorr.	Abbott	P-Value(*)	Rate	Rel. to Control	P-Value(#)
Kontrolle	0.000	3.3	0.0		10.3	0	
Pi3.14159265	0.002	0.0	-3.4	1.000	10.4	-0.6	1.000
Pi3.14159265	0.005	13.3	10.3	1.000	9.9	3.9	0.981
Pi3.14159265	0.010	10.0	6.9	1.000	10.4	-0.6	1.000
Pi3.14159265	0.023	63.3	62.1	<.001	n.d.	n.d.	
Pi3.14159265	0.050	70.0	69.0	<.001	n.d.	n.d.	
*WirktImmer	1.000	100.0	100.0		n.d.	n.d.	

LD50: 0.024 g ai/ha; 95 % Confidence Interval: (0.018 - 0.034)
* Fisher`s Exact test, two-sided, p-values are adjusted according to Bonferroni-Holm
# one-way ANOVA, p-values are adjusted according to Dunnett

Aufbau der Ablage

Aufbau der Ablage

- Pfad mit Versuchs-Nummer erstellen ggf. Version anlegen
  (ältere Berechnungsprotokolle werden nicht überschrieben)
  - \data für SAS und Excel
    - > Excel-Eingabesheet, passwortgeschützt
    - > Zwischenergebnisse aus SAS
  - \docu für Word und Grafiken

LD50-Bestimmung

LD50-Bestimmung

Bei der Bestimmung der LD50 (Dosis die für 50 % der Tiere tötlich ist) ist eine Adjustierung um die natürliche Mortalität notwendig (Abbott Formel).

$p_i^* = \pi + (1-\pi)p_i$

p_i^*: beobachtete Mortalität unter Dosis d_i

π: natürliche Mortalität

p_i: zu schätzende Mortalität unter d_i

Umsetzung im SAS mittels proc probit, Option optc. Es müssen keine speziellen Routinen programmiert werden, welche eine aufwendige Validierung gegen GLP benötigen

Zusammenfassung

Zusammenfassung

- Ziel:
  - neue Guidelines einarbeiten
  - benutzerfreundliche Lösung
    (User arbeitet im MS Office-Umfeld)