

Untersuchung der weltweiten Temperaturgeschichte: Verarbeitung von MATLAB®-Ereignissen in Excel

Mit dem MATLAB Builder for .NET können MATLAB-Programme in Desktopanwendungen wie Excel eingebaut werden. Auf diese Weise lassen sich vertraute Programmoberflächen mit den dreidimensionalen Analyse- und Visualisierungsfähigkeiten von MATLAB ausstatten. Excel kann dadurch beispielsweise auf Ereignisse reagieren, die in einem MATLAB-basierten GUI auftreten. Mit dem MATLAB Builder for .NET erzeugte COM-Komponenten können uneingeschränkt an Anwender weitergegeben werden, die selber über keine MATLAB-Installation verfügen.

Die ‚Lingua Franca‘ für viele Microsoft-Anwendungen, darunter auch Excel, ist Visual Basic. In einer eingebetteten Visual Basic-Umgebung sind nur vier Schritte nötig, um mit dem MATLAB Builder for .NET eine COM-Komponente, die eine Ereignisquelle enthält, aufzubauen und zu nutzen:

1. Implementierung einer Ereignisquelle mit M-Code, der dann zur Erzeugung einer COM-Komponente dient
2. Erzeugung einer Instanz dieser COM-Komponente in Visual Basic
3. Erzeugung einer Ereignissenke in Visual Basic
4. Verarbeitung der ereignisbezogenen Daten

In diesem Artikel werden diese Schritte am Beispiel einer Anwendung vorgestellt, die die weltweiten Temperaturaufzeichnungen von 1880 bis 2005 analysiert und fertige MATLAB-Datensätze in Excel-Diagramme überträgt. (Da es in diesem Artikel vor allem um die Integration in Excel geht, ist der Programmcode überwiegend in Visual Basic verfasst; nur der Abschnitt „Implementierung einer Ereignisquelle“ enthält M-Code.)

Die Anwendung besteht aus zwei Teilen: einer mit MATLAB-Code erzeugten COM-Komponente, die den Datensatz einliest, das Navigations-GUI anzeigt und die Diagramm Daten berechnet, sowie einem von Excel bereitgestellten Visual Basic-Programm, das Diagramme anzeigt, deren Inhalt durch die Ereignisse im GUI bestimmt wird.

Implementierung der Ereignisquelle

Eine fertige COM-Komponente exportiert standardmäßig eine Funktion für jedes kompilierte M-File. Damit eine COM-Komponente eine Ereignisquelle erzeugen kann, muss das verwendete Call-back-M-File das Schlüsselwort `%#event` enthalten. Ein Beispiel:

```
function PlotTempData(city, xData, localAnomData,...
                    fitval, latStr, lngStr)
%#event
```

In MATLAB stellt die Funktion `PlotTempData` die Eingabedaten grafisch dar. In der kompilierten Fassung ändert sich ihr Verhalten aber durch das Schlüsselwort `%#event`. Wenn man in der generierten COM-Komponente die Funktion `PlotTempData` ausführt, erzeugt diese das Ereignis `PlotTempData`, anstatt den Programmcode des M-Files abzurufen. Funktionen für Ereignisquellen müssen keinen M-Code enthalten. Sollte doch M-Code darin enthalten sein, dann wird er nur von MATLAB ausgeführt, nicht jedoch von der fertigen COM-Komponente.

Mit dem MATLAB Builder for .NET wird nun aus den M-Funktionen `GlobalTemp`, `DisplayLocation` und `PlotTempData` eine COM-Komponente erzeugt, die außerdem die Datenfiles `Cities.csv` und `TempAnomData1880.mat` enthält:

```
mcc -g -W com:GlobalTemp,GlobalTempClass,1.0 ...
-T link:lib GlobalTemp DisplayLocation PlotTempData
-a TempAnomData1880.mat -a Cities.csv
```

Die so erzeugte Komponente namens `GlobalTemp` exportiert ihre API mit Hilfe der Klasse `GlobalTempClass`.

Erzeugung einer Instanz der COM-Komponente

Zur Erzeugung einer Instanz einer COM-Komponente muss man eine Referenzvariable deklarieren und initialisieren. Um dabei das Schlüsselwort `WithEvents` verwenden zu können, das eine Komponente als Ereignisquelle kennzeichnet, muss man diese Referenzvariable innerhalb einer Visual Basic-Klasse deklarieren.

In der folgenden Deklaration stellt `tempGUI` eine Instanzvariable der Klasse `TempAnomaly` dar:

```
Dim WithEvents tempGUI as GlobalTemp.GlobalTempClass
```

Die Referenzvariable wird durch die Klassen-Funktion `Class _ Initialize()` initialisiert. Visual Basic ruft diese Funktion jedes Mal auf, wenn es eine neue Instanz der Klasse erzeugt.

```
Private Sub Class_Initialize()
    Set tempGUI = New GlobalTemp.GlobalTempClass
End Sub
```

Erzeugung der Ereignissenke

Eine Ereignissenke ist ein Objekt mit einer Callback-Funktion. Um auf ein Ereignis zu warten, das von einer MATLAB-Ereignisquelle erzeugt wird, muss die Callback-Funktion der Senke die gleiche Signatur tragen wie die zugehörige MATLAB-Funktion. Die zur Klasse `TempAnomaly` gehörende Funktion `tempGUI_PlotTempData` benötigt beispielsweise die gleiche Zahl und Art von Argumenten wie die MATLAB-Funktion `PlotTempData`:

```
Private Sub tempGUI_PlotTempData(ByVal city As Variant, _
    ByVal xData As Variant, ByVal localAnomData As _
    Variant, ByVal fitval As Variant, ByVal latStr _
    As Variant, ByVal lngStr As Variant)
    ' Event handling code goes here
End Sub
```

Vorsicht: Wenn eine mit dem MATLAB Builder for .NET erzeugte COM-Komponente ein Ereignis erzeugt, wartet sie auf die Beendigung der Ereignisverarbeitung, bevor sie selber weiter ausgeführt wird. Callback-Funktionen dürfen also keine Funktionen aufrufen, die von der Komponente selber exportiert werden, da sich sonst eine Endlosschleife bildet.

Verarbeitung ereignisbezogener Daten

Das GUI der COM-Komponente wird in Excel gestartet, indem die in die COM-Komponente eingebettete Funktion `GlobalTemp` aufgerufen wird.

```
Public Sub LaunchGUI()
    tempGUI.GlobalTemp
End Sub
```

Der verwendete Datensatz besteht aus 2592 nach Längengraden geordneten Datenreihen im Monatsabstand, von denen jeder einem fünf mal fünf Grad großen Gebiet auf der Erde entspricht. Die Daten stammen vom National Climatic Data Center der USA. Das GUI stellt die Daten auf einem rotierenden dreidimensionalen Globus dar (Abb. 1). Der Anwender kann daraus das gewünschte Gebiet sehr intuitiv auswählen. Die Datenreihen werden als Streuplot mit Regressionslinien dargestellt, was den Vergleich zwischen verschiedenen Orten vereinfacht. Jedes `PlotTempData`-Ereignis liefert sechs MATLAB-Datensätze an die Ereignissenke in Visual Basic.

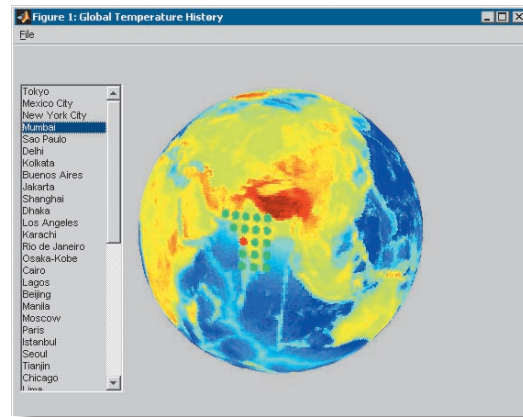


Abb. 1: Schnittstelle für die Datenavigation.

Der Excel-basierte Visual Basic-Code übergibt diese Daten zur Verarbeitung an ein Arbeitsblatt und erzeugt daraus ein Diagramm.

Transfer der MATLAB-Daten in das Arbeitsblatt

Als Reaktion auf das `PlotTempData`-Ereignis überträgt die Callback-Funktion `tempGUI _ PlotTempData` die Ereignisdaten aus den MATLAB-Variablen in das aktive Excel-Arbeitsblatt. Da MATLAB die Daten in Form zweidimensionaler Varianten an die Callback-Funktion übergibt, extrahiert `tempGUI _ PlotTempData` die x-Werte aus dem `xData`-Zellarray durch Varianten-Indexierung.

```
Dim xValues As Variant
xValues = xData(1, 1)
```

Excel kann lange Datenreihen nur grafisch darstellen, wenn sie in ein Arbeitsblatt eingefügt wurden. Im vorliegenden Beispiel haben die Arrays der X- und Y-Werte die gleiche Länge. Alle drei Datenreihen können also mit Hilfe der gleichen `For`-Schleifen extrahiert und in das Arbeitsblatt kopiert werden. Die von MATLAB verwendete Eins-basierte Indexierung ist identisch mit der Eins-basierten Indexierung der Bereiche in Excel-Arbeitsblättern.

```
For rowCount = 1 To UBound(localAnomData, 1)
    For ColCount = 1 To UBound(localAnomData, 2)
        ActiveSheet.Cells(index, tempCol) = _
            localAnomData(rowCount, _ColCount)
        ActiveSheet.Cells(index, fitCol) = _
            fitval(rowCount, ColCount)
        ActiveSheet.Cells(index, dateCol) = _
            xValues(1, index - startRow + 1)
        index = index + 1
    Next ColCount
Next rowCount
```

Danach erzeugt die Callback-Funktion ein Diagramm aus dem Arbeitsblatt, indem sie die private Methode `CreateChart` aufruft.

Erzeugung der Excel-Diagramme

Weil Excel keine Daten vor dem 1. Januar 1900 verarbeiten kann, müssen wir die x-Achse der Zeitreihen-Streuplots selber beschriften (Abb. 2).

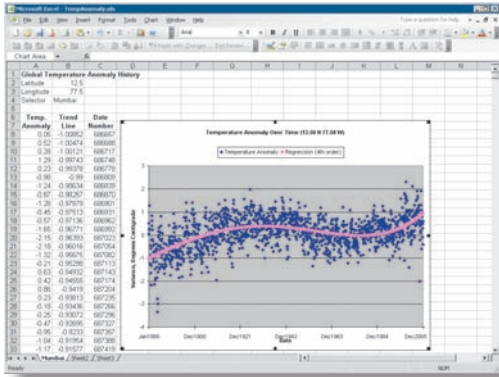


Abb. 2: Temperaturdaten und Regressionslinie für ein einzelnes Gitterelement (Microsoft Excel).

Der unten gezeigte Code fügt die erste Datenreihe in das Diagramm ein. Darin ist `tempRange` der Excelbereich mit den Temperaturdaten und `dateRange` der Bereich mit dem Aufzeichnungsdatum jedes Temperaturwerts.

```
chartObj.chart.SeriesCollection.Add
Source:=tempRange
With chartObj.chart.SeriesCollection(1)
    .Name = "Temperature Anomaly"
    .xValues = dateRange
End With
```

Die COM-Komponente berechnet zunächst sieben Skalenstriche sowie deren Beschriftung auf der x-Achse und gibt diese als MATLAB-Matrix an die Ereignissenke weiter. Diese Daten müssen von einer zweidimensionalen (1x1, 1x7)-Matrix in ein eindimensionales, nullbasiertes (1x7)-Array umgewandelt werden, damit Excel sie darstellen kann.

```
ReDim xLabelData(UBound(xLabels(1, 1), 2) - 1) As
Double
For k = LBound(xLabels(1, 1), 2) To UBound(xLabels(1, 1), 2)
    xLabelData(k - 1) = xLabels(1, 1)(1, k)
Next k
```

Excel unterstützt zwar nicht direkt selbst erzeugte Skalenstriche und -beschriftungen; dies lässt sich aber umgehen, indem wir sie wie eine weitere Datenreihe behandeln. Die als Skalenstrich fungierenden Datenpunkte werden als "+"-Zeichen angezeigt. Die folgende For-Schleife weist jedem Skalenstrich die passende Datumsbeschriftung zu und sorgt dafür, dass sie unterhalb der x-Achse angezeigt wird.

Glossar zur Ereignisverwaltung

Component Object Model (Com). Ein objektorientiertes Programmiermodell, das festlegt, wie Objekte innerhalb einer Anwendung oder über Anwendungen hinweg miteinander wechselwirken.

Ereignis. Ein Vorgang, der eine Aktion auslöst (etwa ein Mausklick).

Ereignisquelle. Eine Komponente, die Ereignisse erzeugt.

Ereignissenke. Eine Komponente, die auf Ereignisse wartet.

Callback-Funktion. Eine Funktion, die einer Softwarekomponenten mitteilt, dass ein Ereignis stattgefunden hat.

Ereignismodell. Gesamtheit der Regeln, die das Verhalten von Ereignissen bestimmen.

```
With chartObj.chart.SeriesCollection(3)
    For k = 1 To .Points.Count
        .Points(k).HasDataLabel = True
        With .Points(k).DataLabel
            .Text = " " + xlabelStr(k, 1) + " "
            .NumberFormat = "Text"
            .Type = xlDataLabelsShowLabel
            .Position = xlLabelPositionBelow
        End With
    Next k
End With
```

Der Ereignis-Handler `PlotTempData` erzeugt alle Datenreihen und Diagramme im jeweils aktiven Tabellenblatt. Man kann also mehrere Diagramme in die gleiche Excel-Arbeitsmappe einfügen, indem man ein neues Blatt öffnet, bevor man ein neues Gitterkästchen im Navigations-GUI auswählt.

Diese Anwendung nutzt sowohl die Stärken von MATLAB als auch die von Excel: Das Navigations-GUI wirkt durch MATLABs dreidimensionale Grafiken professioneller und durch die Tabellenreiter in den Excel-Arbeitsmappen lassen sich viele Datenreihen nebeneinander anzeigen und vergleichen. Das entstandene Programm ist effizienter, wartungsfreundlicher und leichter zu bedienen als eine Anwendung, die mit nur einem dieser Werkzeuge erstellt worden wäre. Zudem kann jeder Excel-Anwender diese Anwendung nutzen, weil mit dem MATLAB Builder for .NET erzeugte Anwendungen uneingeschränkt an andere weiter gegeben werden dürfen. ◀◀

QUELLEN

- ▶ **Overview of the Data Set**
www.ncdc.noaa.gov/oa/climate/research/ghcn/ghcngrid.html
- ▶ **Global Temperature Analysis Code**
www.mathworks.com/res/temperature

