TV12 (Talk vor 12):

Zeitreihenanalyse mit Mathematica und dem Paket MKToolsMM7.m

von

Marko Kastens

Historie / Philosophie

- *die erste Version der Bibliothek entstand 2004*
- *immer wiederkehrende Aufgaben → Migration von Funktionen in die Bibliothek*
- *genetischer Ansatz: nur die Funktionen überleben (den Releasewechsel), die auch gebraucht werden*

Warum Mathematica ? → Was ist Mathematica ?

- *Algebrasystem*
- *rapid prototyping system*
- *Skriptsprache*
- *Visualisierungssystem*
- *Entwicklungssystem*

```
In[1]:= Integrate[1 / (x^3 + 1), x]

Out[1]= ArcTan[(-1+2 x)/Sqrt[3]]/Sqrt[3] + 1/3 Log[1 + x] - 1/6 Log[1 - x + x^2]
```

Es bietet eine Vielzahl an Algorithmen und mathematischen Methoden mit sehr guter Dokumentation!

Mehr Infos:

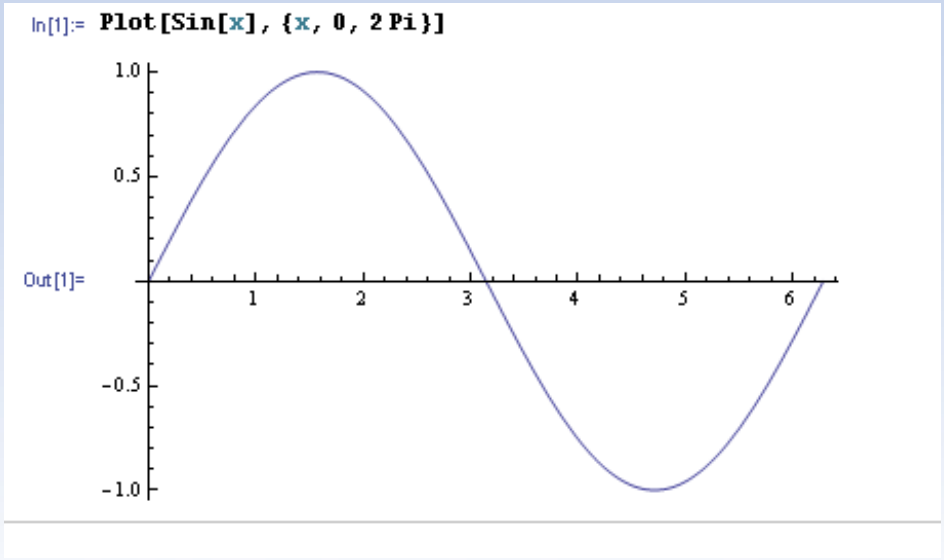
http://www.wolfram.com/products/mathematica/index.html

http://mathworld.wolfram.com

→ Extrem hohe Flexibilität, sehr gute Dokumentation mit vielen Beispielen, sehr gute Performance, unendliche Möglichkeiten

Wie funktioniert Mathematica ?

1. *Notebook(Datei *.nb) laden oder neue öffnen*
2. *Befehl(e) eingeben*
3. *Shift+Return → Übergabe an den Kernel zur Berechnung*
4. *Ergebnisdarstellung*



Ergebnisse bleiben im Speicher (Debug-Zustand).

Die Bibliothek MKToolMM7

- *Aktualisiert auf Version Mathematica Version 7*
- *Größe: 208 kB (ca. 3500 Zeilen)*
- *Umfasst 80 Funktionen*
- *rudimentäre Dokumentation*

Paket in das aktuelle Notebook laden:

<< „**D:\\ Analyse \\ MKToolsMM7_log.m**“

ab jetzt live…

Einführungsbeispiel - Synthese, Kennwerte, Darstellung

In[1]:=
```
(* Bibliothek laden *)
<< "D:\\Analyse\\MKToolsMM7_log.m"
```

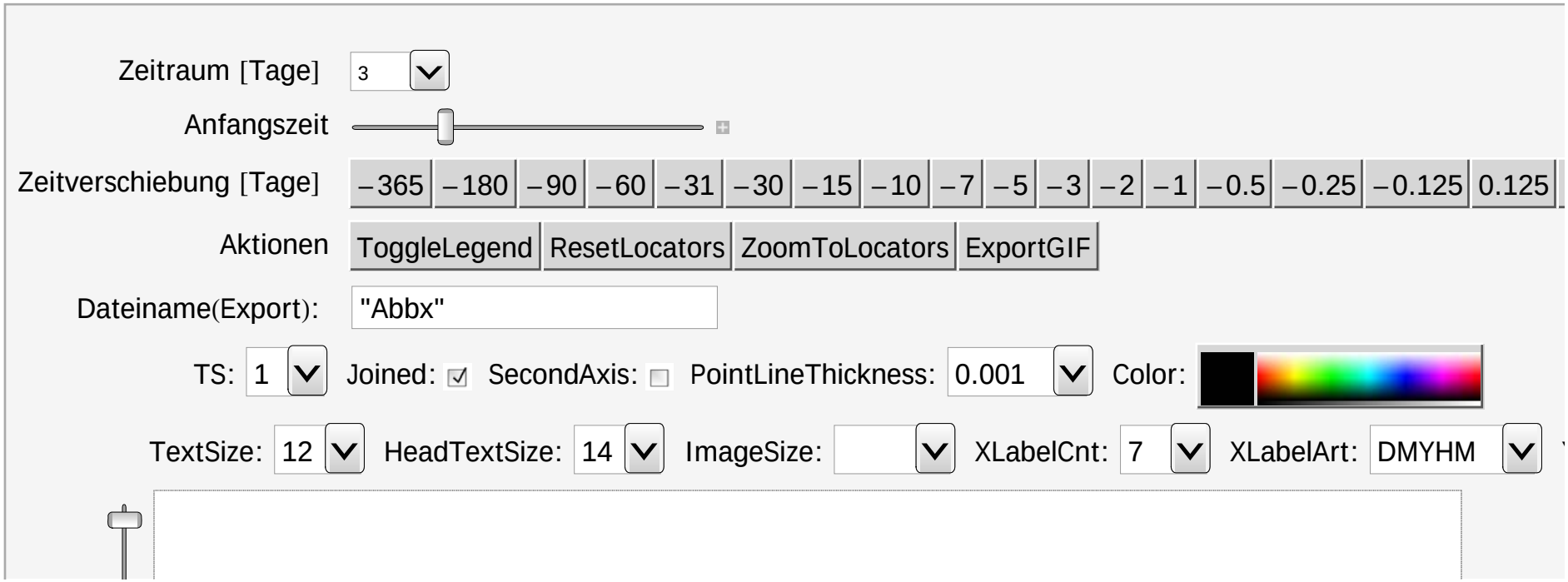
In[2]:=
```
TS = PTSynthese[{{"M_2", 100, 0}, {"S_2", 20, 30}, {"K_1", 5, 130}},
    AbsoluteTime[{2000, 1, 1, 0, 0, 0}], Anfangszeit -> AbsoluteTime[{2000, 1, 1, 0, 0, 0}],
    Endzeit -> AbsoluteTime[{2000, 3, 31, 23, 59, 59}]];
TSFrame[TS]
TableForm[TSHT[TS]]
```

Out[3]=
```
01.01.2000 00:00:00 - 31.03.2000 23:30:00 --> 4368 Datensätze. Länge: 0a90d23h30m0s
```

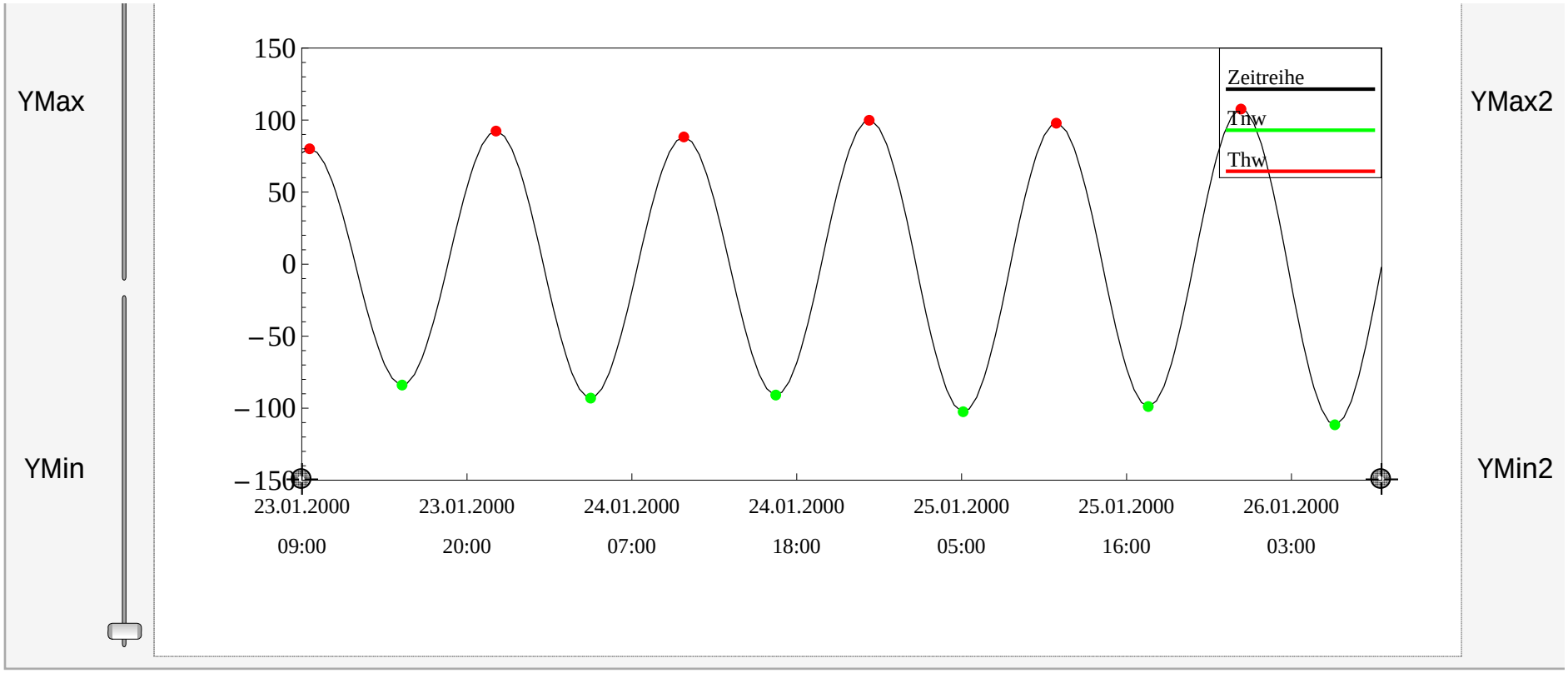
Out[4]//TableForm=

01.01.2000 00:00:00	13.8302	31.03.2000 19:00:00	25.6777
01.01.2000 00:30:00	42.5433	31.03.2000 19:30:00	-1.07884
01.01.2000 01:00:00	68.6421	31.03.2000 20:00:00	-27.5705
01.01.2000 01:30:00	90.4278	31.03.2000 20:30:00	-52.1363
01.01.2000 02:00:00	106.479	31.03.2000 21:00:00	-73.2452
01.01.2000 02:30:00	115.743	31.03.2000 21:30:00	-89.593
01.01.2000 03:00:00	117.604	31.03.2000 22:00:00	-100.184
01.01.2000 03:30:00	111.923	31.03.2000 22:30:00	-104.394
01.01.2000 04:00:00	99.0456	31.03.2000 23:00:00	-102.005
01.01.2000 04:30:00	79.7791	31.03.2000 23:30:00	-93.2222

In[5]:=
```
{Tnw, Thw} = TSScheitelFull[TS];
TSListPlotArrayI[{TS, Tnw, Thw}, JoinedList → {True, False, False},
 ColorList → {Black, Green, Red}, PointLineThickness → {0.001, 0.01, 0.01},
 YRange → {-150, 150}, YRange2 → {-120, 120}, ISize → 700,
 LegendEntryList → {"Zeitreihe", "Tnw", "Thw"}]
```



Out[6]=


YMax
YMin
YMax2
YMin2
150
100
50
0
−50
−100
−150
Zeitreihe
Thw
Thw
23.01.2000
09:00
23.01.2000
20:00
24.01.2000
07:00
24.01.2000
18:00
25.01.2000
05:00
25.01.2000
16:00
26.01.2000
03:00

◄ | ►

Informationen & Optionen

In[7]:=
```
?PTSynthese
?TSFrame
?TSHT
```

PTSynthese[Partialtidenliste, Referenzzeitpunkt, Optionen] –> Zeitreihe(TS)
Partialtidenliste = {{PTName 1, Amplitude 1, Phase 1}, ...}
Die Funktion erzeugt eine Zeitreihe aus den Partialtiden der Partialtidenliste. Die Phase der Partialtide bezieht sich
auf den angegebenen Referenzzeitpunkt. Die Zeitspanne der Zeitreihe kann über die Optionen Anfangs– und Endzeit frei gewählt werden.

Optionen:
Anfangszeit –> AbsoluteTime[{2000, 1, 1, 0, 0, 0}]
Endzeit –> AbsoluteTime[{2000, 12, 31, 23, 30, 0}]
Sampledauer –> 30*60 (Zeitabstand zweier Datenpunkte)
VerboseMode –> False (True –> mehr Infos)

TSFrame[TS, Optionen] –> String
TSFrame gibt den Rahmen einer Zeitreihe(TS) aus: erster Datenpunkt, letzter Datenpunkt,
[Optional: Anzahl der Daten], [Optional: Länge der Zeitreihe], [Optional: Mittelwert und Standardabweichung der Zeitschrittweite]

Optionen:
ShowDatasets –> True (zeigt Anzahl der Datensätze an)
ShowLength –> True (zeigt die Länge der Zeitreihe an)
CalcdT –> False (zeigt Mittelwert und Standardabweichung der Zeitschrittweite an)

TSHT[TS, Optionen] –> Die Funktion gibt die ersten und letzen n Zeilen der Zeitreihe TS aus.

Optionen:
AnzZeilen –> 10 (Anzahl der Zeilen)

Übersicht

- Datenimport / -export
- Zeitreihenfunktionen I
- Darstellung / Abbildungen
- Scheitelwerte
- Zeitreihenfunktion II
- Filter
- Strömungen
- Partialtiden

Datenimport (Boewert)

In[10]:=
```
?ReadBoe
{TS, nfp, name, rw, hw, anzpar, phylist} =
  ReadBoe["D:\\Analyse\\_Library\\Test.boe", VerboseMode → False];
TSFrame[
 TS]
```

ReadBoe[DirectoryundDateiname, Optionen] –> {Zeitreihe TS, Nummer des feuchten Punktes, Name, Rechtswert, Hochwert, Anzahl der Parameter, PhyList}
Die Funktion liest eine BOEWERT–Datei ein und konvertiert sie in ein Mathematica–Zeitreihen–Format[Zeit in Sek nach 1900, Wert1, ..., Wert n].

Optionen:
Trennzeichen –> _; (Übergabe der Trennzeichen, z.B. ;)
ReadSeconds –> False (True/False – liest bzw. überspringt das Einlesen der Sekunden)
VerboseMode –> False (True –> mehr Infos)
DebugMode –> False (True –> DebugInformationen)

Out[12]= `01.01.1997 00:01:00 – 01.01.1997 08:41:00 –-> 53 Datensätze. Länge: 0a0d8h40m0s`

In[13]:=
```
TS
nfp
name
rw
hw
anzpar
phylist
```

Out[13]= {{3061065660, 377.01, -2.}, {3061066260, 393.96, -2.},
{3061066860, 409.34, -2.}, {3061067460, 428.59, -2.}, {3061068060, 446.9, -2.},
{3061068660, 467.43, -2.}, {3061069260, 484.35, -2.}, {3061069860, 501.7, -2.},
{3061070460, 518.08, -2.}, {3061071060, 535.51, -2.}, {3061071660, 551.24, -2.},
{3061072260, 565.27, -2.}, {3061072860, 578.01, -2.}, {3061073460, 590.66, -2.},
{3061074060, 600.87, -2.}, {3061074660, 610.95, -2.}, {3061075260, 618.13, -2.},
{3061075860, 625.32, -2.}, {3061076460, 631.79, -2.}, {3061077060, 636.2, -2.},
{3061077660, 640.1, -2.}, {3061078260, 642.79, -2.}, {3061078860, 645.27, -2.},
{3061079460, 647.85, -2.}, {3061080060, 650.1, -2.}, {3061080660, 651.7, -2.},
{3061081260, 652.41, -2.}, {3061081860, 652.07, -2.}, {3061082460, 650.55, -2.},
{3061083060, 648.85, -2.}, {3061083660, 646.58, -2.}, {3061084260, 644.13, -2.},
{3061084860, 639.45, -2.}, {3061085460, 634.91, -2.}, {3061086060, 627.92, -2.},
{3061086660, 620.75, -2.}, {3061087260, 610.47, -2.}, {3061087860, 600.27, -2.},
{3061088460, 589.5, -2.}, {3061089060, 578.23, -2.}, {3061089660, 566.19, -2.},
{3061090260, 554.46, -2.}, {3061090860, 542.42, -2.}, {3061091460, 529.83, -2.},
{3061092060, 516.87, -2.}, {3061092660, 504.08, -2.}, {3061093260, 491.37, -2.},
{3061093860, 480.05, -2.}, {3061094460, 468.42, -2.}, {3061095060, 457.06, -2.},
{3061095660, 446.01, -2.}, {3061096260, 435.1, -2.}, {3061096860, 424.52, -2.}}

Out[14]= 0

Out[15]= BakeA

Out[16]= 3455144

Out[17]= 5983959

Out[18]= 2

Out[19]= {3, 1397}

Datenexport (Boewert)

In[20]:=
```
?WriteBoe
Directory[]
WriteBoe["Test.dat", TS, 0, "Testdaten", rw, hw, phylist];
```

WriteBoe[DirectoryundDateiname, Zeitreihe TS, Numer des feuchten Punktes, Name, Rechtswert, Hochwert, Phycodes, Optionen]
Die Datei exportiert eine Zeitreihe TS in das BOEWERT−Format.

Optionen:
NKStellenListe −> Automatic (oder Liste mit Nachkommastellen die der Reihenfolge der PHYDEF−Codes entsprechen)

Out[21]= D:\Eigene Dateien

Datenimport/-export

In[199]:=
```
(* Mathematica-Funktionen: Import/Export, ReadList/WriteList, Get/Put, ... *)
tmpTS = Get[
    "D:\\Pegel\\Daten\\Deutschland\\Elbe\\_10minDaten\\passed\\nur2006\\BakeA_01012006-
      31122006.mm"]; // AbsoluteTiming
TSFrame[tmpTS]
Dimensions[tmpTS]
TSlong = tmpTS[[All, {1, 2}]];
```

Out[199]= {0.2812392, Null}

Out[200]= 01.01.2006 00:01:00 - 31.12.2006 23:51:00 --> 52560 Datensätze. Länge: 0a364d23h50m0s

Out[201]= {52560, 4}

Zeitreihenfunktionen I - TSCheck

In[27]:=
```
? TSCheck
TSCheck[TS]
```

TSCheck[TS] –> True/False
Testet, ob TS eine Zeitreihe ist.
Voraussetzung ist: eine Arraytiefe von genau 2 mit mindestens einer Datenzeile. Bei unterschiedlich dimensionalen Zeitreihen wird False und ggf. eine Warnung ausgegeben.

Optionen:
VerboseMode –> True (False –> Warnungen werden nicht ausgegeben)

Out[28]= `True`

Zeitreihenfunktionen I - TSFrame

In[29]:=
```
?TSFrame
TSFrame[TS]
```

TSFrame[TS, Optionen] –> String
TSFrame gibt den Rahmen einer Zeitreihe(TS) aus: erster Datenpunkt, letzter Datenpunkt,
[Optional: Anzahl der Daten], [Optional: Länge der Zeitreihe], [Optional: Mittelwert und Standardabweichung der Zeitschrittweite]

Optionen:
ShowDatasets –> True (zeigt Anzahl der Datensätze an)
ShowLength –> True (zeigt die Länge der Zeitreihe an)
CalcdT –> False (zeigt Mittelwert und Standardabweichung der Zeitschrittweite an)

Out[30]=
```
01.01.1997 00:01:00 - 01.01.1997 08:41:00 --> 53 Datensätze. Länge: 0a0d8h40m0s
```

Zeitreihenfunktionen I - TSHT

In[31]:=
```
?TSHT
TableForm[TSHT[TS]]
```

TSHT[TS, Optionen] –> Die Funktion gibt die ersten und letzen n Zeilen der Zeitreihe TS aus.

Optionen:
AnzZeilen –> 10 (Anzahl der Zeilen)

Out[32]//TableForm=

01.01.1997	00:01:00	377.01	–2.	01.01.1997	07:11:00	529.83	–2.
01.01.1997	00:11:00	393.96	–2.	01.01.1997	07:21:00	516.87	–2.
01.01.1997	00:21:00	409.34	–2.	01.01.1997	07:31:00	504.08	–2.
01.01.1997	00:31:00	428.59	–2.	01.01.1997	07:41:00	491.37	–2.
01.01.1997	00:41:00	446.9	–2.	01.01.1997	07:51:00	480.05	–2.
01.01.1997	00:51:00	467.43	–2.	01.01.1997	08:01:00	468.42	–2.
01.01.1997	01:01:00	484.35	–2.	01.01.1997	08:11:00	457.06	–2.
01.01.1997	01:11:00	501.7	–2.	01.01.1997	08:21:00	446.01	–2.
01.01.1997	01:21:00	518.08	–2.	01.01.1997	08:31:00	435.1	–2.
01.01.1997	01:31:00	535.51	–2.	01.01.1997	08:41:00	424.52	–2.

Zeitreihenfunktionen I - TSInfo

In[33]:=
```
?TSInfo
TSInfo[TS];
```

TSInfo[TS, Optionen] –> Infos
Die Funktion gibt allgemeine Informationen über eine Zeitreihe aus.

Optionen:
FrameInfo –> True (zeigt Informationen über Anfang, Ende, Länge und Anzahl der Datensätze)
ShowDim –> True (zeigt die Dimensionen der Zeitreihe an)
CalcMeans –> True (berechnet Mittelwert, Standardabweichung, Minimum und Maximum für Zeitabstände und Werte der Zeitreihe)
ListdTChange –> True (zeigt Veränderungen im zeitlichen Abstand der Datensätze)
ListBadValues –> True (zeigt die Anzahl der BadValues an)
BadValues –> {0., –777.} (Liste mit BadValues)

01.01.1997 00:01:00 – 01.01.1997 08:41:00 –-> 53 Datensätze. Länge: 0a0d8h40m0s

Dimensionen: {53, 3}

	Mittel	Stadardabweichung	Min	Max
Spalte 1	600.	0.	600	600
Spalte 2	559.653	83.1443	377.01	652.41
Spalte 3	–2.	0.	–2.	–2.

dt [sec]	von	bis	Anzahl der Datensätze	von Pos.	bis Pos.
600	01.01.1997 00:01:00	01.01.1997 08:41:00	53	1	53

	0.	–777.
Fehlwerte Spalte 2	0	0
Fehlwerte Spalte 3	0	0

Zeitreihenfunktionen I - TSencode

In[35]:= **? TSencode**

TSencode [TS]

TSencode[TS, Optionen] –> Liste
TSencode liefert eine Liste/Tabelle, in der die erste Spalte der Zeitreihe TS (Datum und Zeit) als lesbares Format eingetragen ist. Die weiteren Spalten bleiben erhalten.

Optionen:
DateTimeFormat–>DMYHMS (Möglichkeiten siehe unter DTenc)

Out[36]=
```
{{01.01.1997 00:01:00, 377.01, -2.},
 {01.01.1997 00:11:00, 393.96, -2.}, {01.01.1997 00:21:00, 409.34, -2.},
 {01.01.1997 00:31:00, 428.59, -2.}, {01.01.1997 00:41:00, 446.9, -2.},
 {01.01.1997 00:51:00, 467.43, -2.}, {01.01.1997 01:01:00, 484.35, -2.},
 {01.01.1997 01:11:00, 501.7, -2.}, {01.01.1997 01:21:00, 518.08, -2.},
 {01.01.1997 01:31:00, 535.51, -2.}, {01.01.1997 01:41:00, 551.24, -2.},
 {01.01.1997 01:51:00, 565.27, -2.}, {01.01.1997 02:01:00, 578.01, -2.},
 {01.01.1997 02:11:00, 590.66, -2.}, {01.01.1997 02:21:00, 600.87, -2.},
 {01.01.1997 02:31:00, 610.95, -2.}, {01.01.1997 02:41:00, 618.13, -2.},
 {01.01.1997 02:51:00, 625.32, -2.}, {01.01.1997 03:01:00, 631.79, -2.},
```

```
{01.01.1997 03:11:00, 636.2, -2.}, {01.01.1997 03:21:00, 640.1, -2.},
{01.01.1997 03:31:00, 642.79, -2.}, {01.01.1997 03:41:00, 645.27, -2.},
{01.01.1997 03:51:00, 647.85, -2.}, {01.01.1997 04:01:00, 650.1, -2.},
{01.01.1997 04:11:00, 651.7, -2.}, {01.01.1997 04:21:00, 652.41, -2.},
{01.01.1997 04:31:00, 652.07, -2.}, {01.01.1997 04:41:00, 650.55, -2.},
{01.01.1997 04:51:00, 648.85, -2.}, {01.01.1997 05:01:00, 646.58, -2.},
{01.01.1997 05:11:00, 644.13, -2.}, {01.01.1997 05:21:00, 639.45, -2.},
{01.01.1997 05:31:00, 634.91, -2.}, {01.01.1997 05:41:00, 627.92, -2.},
{01.01.1997 05:51:00, 620.75, -2.}, {01.01.1997 06:01:00, 610.47, -2.},
{01.01.1997 06:11:00, 600.27, -2.}, {01.01.1997 06:21:00, 589.5, -2.},
{01.01.1997 06:31:00, 578.23, -2.}, {01.01.1997 06:41:00, 566.19, -2.},
{01.01.1997 06:51:00, 554.46, -2.}, {01.01.1997 07:01:00, 542.42, -2.},
{01.01.1997 07:11:00, 529.83, -2.}, {01.01.1997 07:21:00, 516.87, -2.},
{01.01.1997 07:31:00, 504.08, -2.}, {01.01.1997 07:41:00, 491.37, -2.},
{01.01.1997 07:51:00, 480.05, -2.}, {01.01.1997 08:01:00, 468.42, -2.},
{01.01.1997 08:11:00, 457.06, -2.}, {01.01.1997 08:21:00, 446.01, -2.},
{01.01.1997 08:31:00, 435.1, -2.}, {01.01.1997 08:41:00, 424.52, -2.}}
```

Zeitreihenfunktionen I - TSSelect

In[37]:=
```
?TSSelect
tmpTS = TSSelect[TS, AbsoluteTime[{1997, 1, 1, 2, 0, 0}], AbsoluteTime[{1997, 1, 1, 5, 1, 0}]]
TSencode[tmpTS]
```

TSSelect[TS, von, bis, Optionen] –> Zeitreihe
Die Funktion zerlegt eine Zeitreihe TS in dem vorgegebenen Zeitraum(von bis) und liefert den Ausschnitt als neue Zeitreihe zurück. Da bei einer aequidistanten Zeitreihe eine andere Methode mit hoher Performance angewandt werden kann, wird zuvor ein Aequidistanzcheck durchgeführt. Dieser wiederum wird aus Performancegründen nur auf Zeitreihen mit weniger als 20.000 Datensätzen angewendet. Zeitreihen, die mehr als 20.000 Datensätze beinhalten, werden nur an 10 Stichproben getestet. Steht die Option CheckAE auf True, wird IMMER ein Aequidistanzcheck durchgeführt.

Optionen:
CheckAE –> False (True –> Testet auf Äquidistanz)
VerboseMode –> False (True –> mehr Infos)

Out[38]=
```
{{3061072860, 578.01, -2.}, {3061073460, 590.66, -2.},
 {3061074060, 600.87, -2.}, {3061074660, 610.95, -2.},
 {3061075260, 618.13, -2.}, {3061075860, 625.32, -2.}, {3061076460, 631.79, -2.},
 {3061077060, 636.2, -2.}, {3061077660, 640.1, -2.}, {3061078260, 642.79, -2.},
 {3061078860, 645.27, -2.}, {3061079460, 647.85, -2.}, {3061080060, 650.1, -2.},
 {3061080660, 651.7, -2.}, {3061081260, 652.41, -2.}, {3061081860, 652.07, -2.},
 {3061082460, 650.55, -2.}, {3061083060, 648.85, -2.}, {3061083660, 646.58, -2.}}
```

Out[39]= {{01.01.1997 02:01:00, 578.01, -2.},
{01.01.1997 02:11:00, 590.66, -2.}, {01.01.1997 02:21:00, 600.87, -2.},
{01.01.1997 02:31:00, 610.95, -2.}, {01.01.1997 02:41:00, 618.13, -2.},
{01.01.1997 02:51:00, 625.32, -2.}, {01.01.1997 03:01:00, 631.79, -2.},
{01.01.1997 03:11:00, 636.2, -2.}, {01.01.1997 03:21:00, 640.1, -2.},
{01.01.1997 03:31:00, 642.79, -2.}, {01.01.1997 03:41:00, 645.27, -2.},
{01.01.1997 03:51:00, 647.85, -2.}, {01.01.1997 04:01:00, 650.1, -2.},
{01.01.1997 04:11:00, 651.7, -2.}, {01.01.1997 04:21:00, 652.41, -2.},
{01.01.1997 04:31:00, 652.07, -2.}, {01.01.1997 04:41:00, 650.55, -2.},
{01.01.1997 04:51:00, 648.85, -2.}, {01.01.1997 05:01:00, 646.58, -2.}}

Zeitreihenfunktionen I - TSCheckAequidistanz

In[40]:=
```
?TSCheckAequidistanz
TSCheckAequidistanz[TS]
TSCheckAequidistanz[Drop[TS, {10}]]
```

TSCheckAequidistanz[TS] –> dt oder bei nicht Äquidistanz {dt1, dt2, ...}
Bei aequidistanten Zeitreihen wird der Abstand zweier Samples in Sekunden (dt) zurückgegeben. Bei nicht–aequidistanten
Zeitreihen wird eine Warnmeldung ausgegeben (VerboseMode –> True, sonst nicht) und eine Liste der verschiedenen Zeitabstände in Sekunden.

Optionen:
VerboseMode –> True (False –> keine Warnungen)

Out[41]=
```
600
```

TSCheckAequidistanz::aequidistantwarning : Die Zeitreihe ist nicht aequidistant!

Out[42]=
```
{600, 1200}
```

Zeitreihenfunktionen I - TSMarkLeapinTime

In[43]:=
```
?TSMarkLeapinTime
badTS = Drop[TS, {20}];
TableForm[TSMarkLeapinTime[badTS]]
```

TSMarkLeapinTime[TS, Optionen] –> StringListe
Die Funktion gibt eine StringListe zurück, in der ein Wechsel des Zeitschritts dokumentiert wird.

Optionen:
TimeStep –> Automatic (erster Zeitschritt, sonst Wert angeben)\,VerboseMode –> False (True –> mehr Infos)

Out[45]//TableForm=

dt [sec]	von	bis	Anzahl der Datensätze	von Pos.	bis Pos.
600	01.01.1997 00:01:00	01.01.1997 03:01:00	19	1	19
1200	01.01.1997 03:01:00	01.01.1997 03:21:00	1	20	20
600	01.01.1997 03:21:00	01.01.1997 08:41:00	32	21	52

Zeitreihenfunktionen I - TSHeal

In[46]:=
```
?TSHeal
Dimensions[badTS]
markedbadTS = TSHeal[badTS, 600];
```

TSHeal[TS, solldt, Optionen] –> TS
Die Funktion ermittelt die Zeitsprünge (<> solldt) in TS. Sind diese vorhanden, wird eine leere Zeitreihe eingesetzt, so daß die zurückgegebene Zeitreihe äquidistant ist.

Optionen:
Emptylist –> Automatic (bei gleichmäßigen Zeitreihen(m x n–Matrix) werden
die neuen Zeilen mit EmptyList(bei Automatic mit Null) gefüllt; anderes Beispiel –> {–1, –1} –> z.B. für Strömung)
VerboseMode –> True (Zwischenergebnisse anzeigen)

Out[47]= {52, 3}

bei leeren Zeilen Werte einfügen: {0, 0}

dt [sec]	von	bis	Anzahl der Datensätze	von Pos.	bis Pos.
600	01.01.1997 00:01:00	01.01.1997 03:01:00	19	1	19
1200	01.01.1997 03:01:00	01.01.1997 03:21:00	1	20	20
600	01.01.1997 03:21:00	01.01.1997 08:41:00	32	21	52

leere Zeitreihe einfügen:
01.01.1997 03:11:00 – 01.01.1997 03:11:00 --> 1 Datensätze. Länge: 0a0d0h0m0s

neue Zeitreihe:
01.01.1997 00:01:00 – 01.01.1997 08:41:00 --> 53 Datensätze. Länge: 0a0d8h40m0s

Zeitreihenfunktionen I - TSDetermineHole

In[49]:=
```
?TSDetermineHole
TSDetermineHole[markedbadTS]
```

TSDetermineHole[TS, Optionen] –> Liste {{von Position, bis Position}, {von Position, bis Position}, ...}
Die Funktion gibt eine Liste mit Anfangs– und Endpositionen zurück, die vollständige
Zeitreihen(ohne badvalue) markieren. Lückendefinitionen werden über die Option BadValues (Default {0, –777}) vorgegeben.

Optionen:
BadValues –> {0., –777.} (Liste mit BadValues)

Out[50]=
```
{{1, 19}, {21, 53}}
```

Zeitreihenfunktionen I - TSStatistik

In[51]:=
```
?TSStatistik
q = TSStatistik[TSlong, Timespace → Month, Mode → Max];
TableForm[TSencode[q]]
TSMW = q[[All, {1, 4}]];
```

TSStatistik[TS, Optionen] –> {{Zeit von, Zeit bis, Anzahl der Werte im Zeitraum, Wert Spalte2, Wert Spalte3, ...}, ...}
Die Funktion zerlegt die Zeitreihe TS in Subzeitreihen je nach Art (Year, Month, Day) und erstellt
Werte je nach Modus [Mean, StandardDeviation, Min, Max] innerhalb dieser Subzeitreihen über alle vorhandenen Spalten von TS.

Optionen:
Mode –> Mean [Mean, StandardDeviation, Min, Max]
Timespace –> Year [Year, Month, Day]
VerboseMode –> False (True –> mehr Infos)

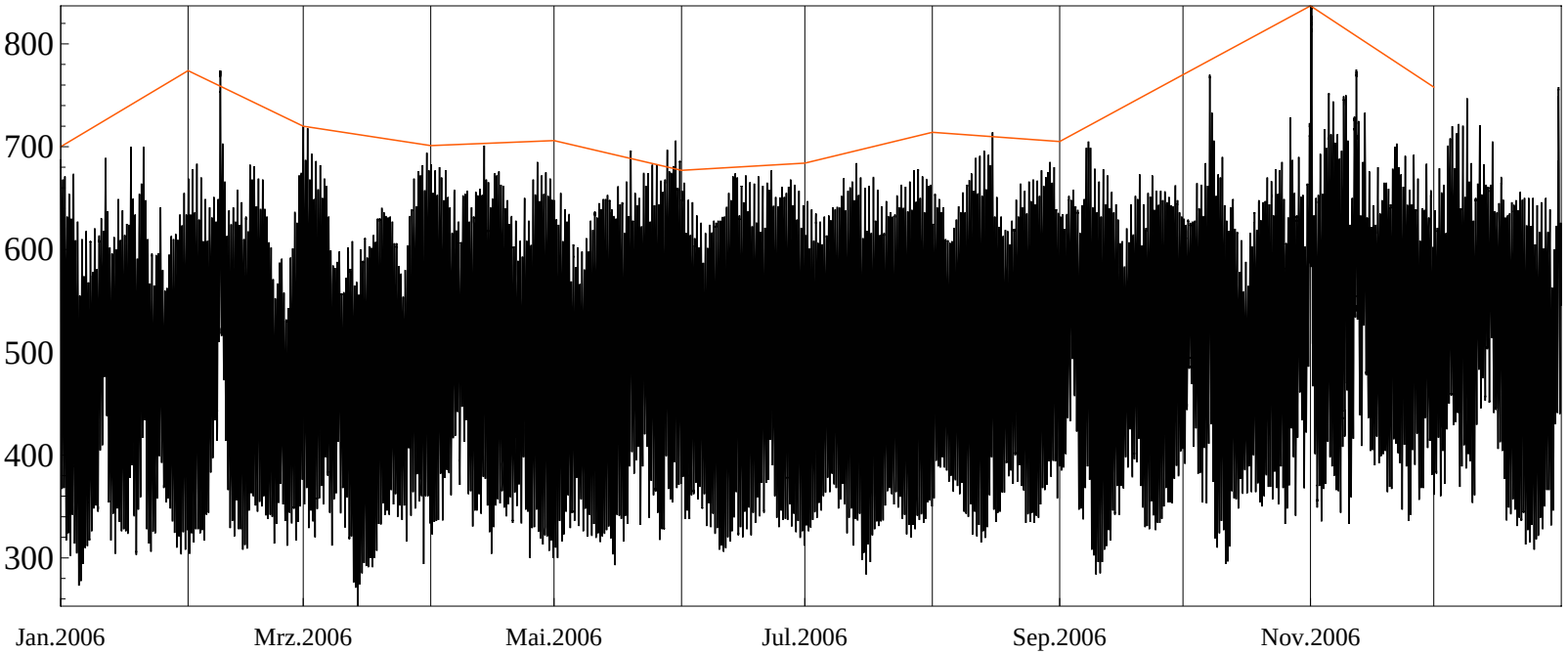
Out[53]//TableForm=

01.01.2006	00:00:00	3 347 740 799	4464	700.
01.02.2006	00:00:00	3 350 159 999	4032	774.
01.03.2006	00:00:00	3 352 838 399	4464	720.
01.04.2006	00:00:00	3 355 430 399	4320	701.
01.05.2006	00:00:00	3 358 108 799	4464	706.
01.06.2006	00:00:00	3 360 700 799	4320	677.
01.07.2006	00:00:00	3 363 379 199	4464	684.
01.08.2006	00:00:00	3 366 057 599	4464	714.
01.09.2006	00:00:00	3 368 649 599	4320	705.
01.10.2006	00:00:00	3 371 327 999	4464	770.
01.11.2006	00:00:00	3 373 919 999	4320	837.
01.12.2006	00:00:00	3 376 598 399	4464	758.

Darstellung (statisch)

In[55]:=
```
TSListPlotArray[{TSlong, TSMW}, Grid -> True,
  GridWhen → Monthly, ISize → 800, Legend → False][[1]]
?TSListPlotArray
```

Out[55]=



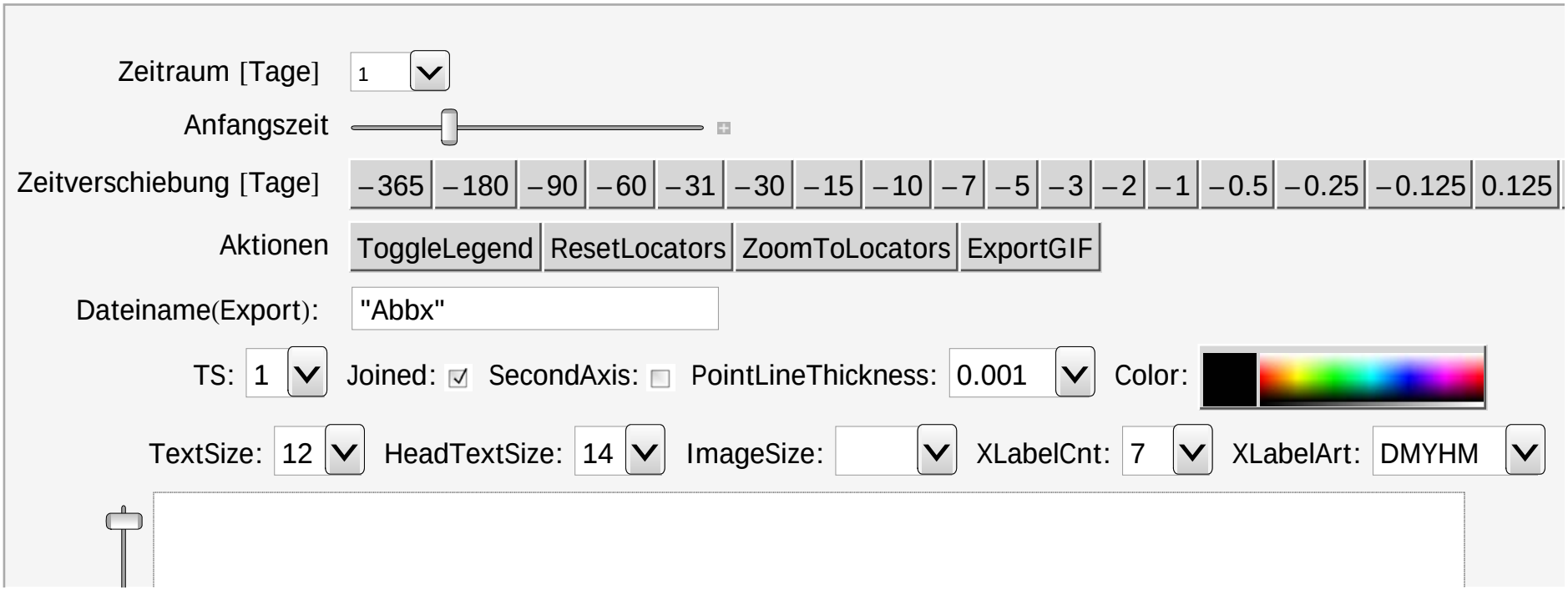
TSListPlotArray[TSArray, Optionen] –> Grafik–Liste
Die Funktion erstellt einen Plot der Zeitreihe(n) in TSArray. TSArray kann sein TS oder {TS1, TS2, ...}

Optionen:
StartTime –> Automatic (Anfangszeit der Darstellung)
EndTime –> Automatic (Endzeit der Darstellung)
IntervalTime –> Automatic (Zeitintervall; ist es kleiner als der
    Gesamtzeitraum, werden soviele Grafiken mit der Darstellungslänge von IntervalTime erzeugt, bis EndTime erreicht ist)
SecondAxisList –> {False, True, ...} (True, wenn Sekundärachse benutzt werden soll)
YRange –> All (Wertebereich der Y–Achse; Automatic, All, Constant(mehrere Plots haben denselben Wertebereich) oder {ymin, ymax)
YRange2 –> All (Wertebereich–Sekundärachse; All oder {ymin, ymax)
NumberYLabelSecondary –> 5 (Anzahl der Labels auf der Sekundärachse)
ColorList –> Automatic (bei Automatic wird eine interne Farbliste verwedet, sonst muss eine Farbliste in der Form {Farbe1, ...} angegeben werden)
JoinedList –> Automatic (Automatic bedeutet für alle Zeitreihen TRUE=verbunden; Alternative: Liste := {True, False, ...})
MarkerList –> Automatic (Automatic oder {Marker1, Marker2, ...} [None –> kein Marker(default), Automatic, oder Objekt]
PointLineThickness –> 0.001 (bei nur einem Wert wird dieser für alle Zeitreihen verwendet; Alternative: Liste := {0.01, 0.001, ...})
PLabel –> (Überschrift)
FontSizeHeader –> 14 (Textgröße der Überschrift)
XLabelAnzahl –> 7 (Anzahl der Beschriftungspunkte auf der X–Achse)
XLabelArt –> Automatic [s. DTenc]
XLabelWhen –> Automatic (genaue Teilung der X–Achse nach XLabelAnzahl[Yearly, Monthly, Daily, Automatic]
FnSize –> 12 (Größe des Resttextes in Pixeln)
Legend –> Automatic [Automatic(True, wenn mehr als eine Zeitreihe vorhanden), True, False]
LegendLabel –> (Überschrift der Legende)
LegendEntryList –> Automatic (Alternativ: Liste := {Eintrag1, ...})

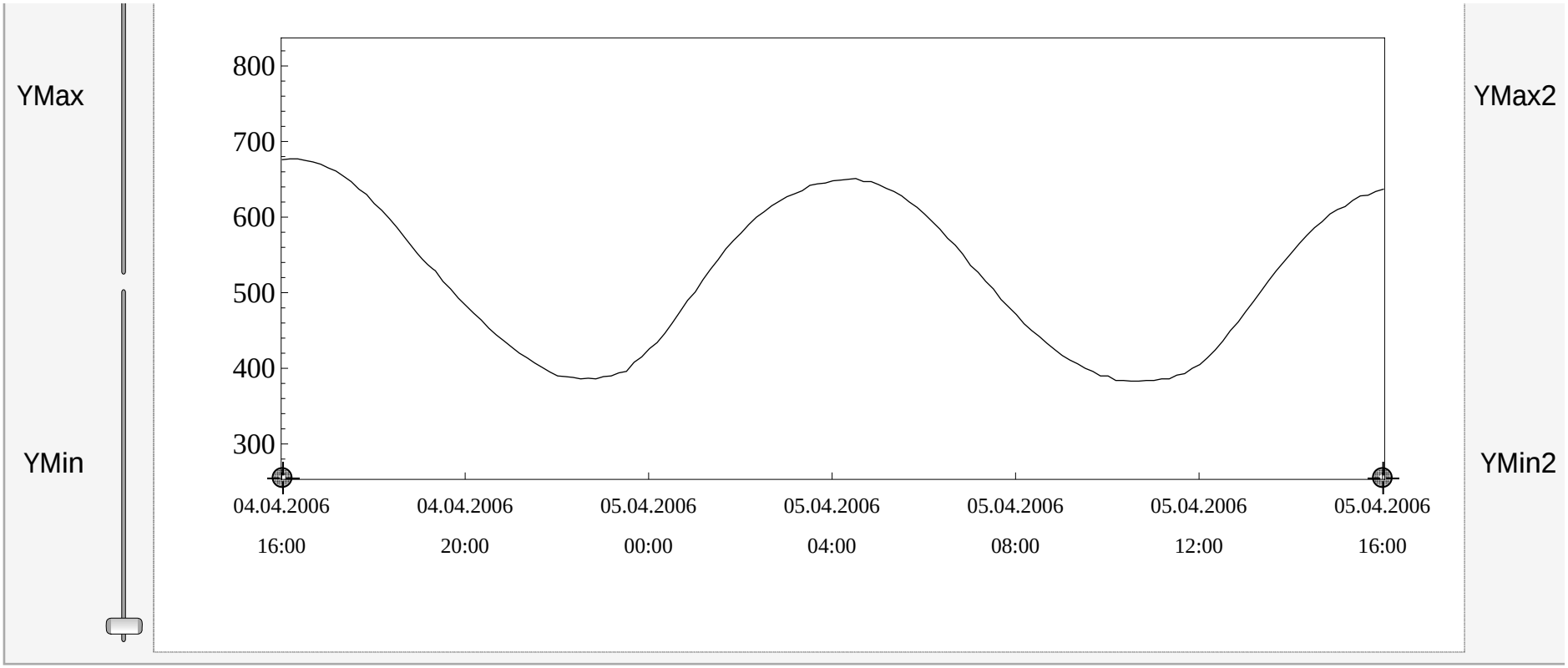
LegendSize –> Automatic (Automatic oder {Breite in %, Höhe in %})
LegendTextSize –> 10 (Textgröße in der Legende)
LabelFrame –> { , , , } (Beschriftung der Frameachsen)
Grid –> False (True zeigt das Grid)
GridWhen –> Daily (wann soll ein Teilstrich gezeichnet werden [Yearly, Monthly, Daily])
GridColor –> RGBColor[0, 0, 0] (die Farbe eines Teilstrichs)
AR –> 2/5 (Breite zu Höhe–Verhältnis)
ISize –> 1000 (Breite der Grafik in Pixeln; Alternative: {Breite, Höhe})
MaxPoints –> 60000 (Maximale Anzahl der Punkte die dargestellt werden; sind mehr vorhanden wird gewarnt und die Zeitreihe wird ausgedünnt)
VerboseMode –> False (True –> mehr Infos)

Darstellung (dynamisch)

In[57]:=
```
TSListPlotArrayI[TSlong, ISize → 700]
?TSListPlotArrayI
```



Out[57]=


YMax
YMin
800
700
600
500
400
300
YMax2
YMin2
04.04.2006
16:00
04.04.2006
20:00
05.04.2006
00:00
05.04.2006
04:00
05.04.2006
08:00
05.04.2006
12:00
05.04.2006
16:00

TSListPlotArrayI[TSArray, Optionen] –> InteraktivesObjekt
Die Funktion erstellt einen Plot der Zeitreihe(n) in TSArray. TSArray kann sein TS oder {TS1, TS2, ...}

Optionen:
YRange –> Automatic (Y–Abschnitt der primären Achse; Bsp.: {ymin, ymax})
YRange2 –> Automatic (Y–Abschnitt der sekundären Achse; Bsp.: {ymin2, ymax2})
ColorList –> Automatic (Farbe für jede Zeitreihe; Bsp: {Black, Blue, Red, ...})
JoinedList –> Automatic (True –> verbunden/False –> Punkte; Bsp.: {True, False, False, ...})
PointLineThickness –> Automatic (Dicke der Punkte, Linien; Bsp.: {0.001, 0.001, 0.005})
SecondAxisList –> Automatic (welche Zeitreihe wird de Sekundärachse zugeordnet; True –> Sekundär; False –> Primär; Bsp.: {True, False)
PLabel –> (Abbildungsüberschrift)
LabelFrame –> { , , , } (Achsbeschriftungen xunten, ylink, xoben, yrechts)
LegendEntryList –> Automatic (oder Liste := {Eintrag1, ...})
WorkDir –> C:\temp\ (Ausgabeverzeichnis, wenn Button ExportGIF gedrückt wird)
DefaultTimeStepDay –> 1/24 (Zeitschrittvorgabe für den Reger 'Anfangszeit')
ISize –> 1000 (Breite der Abbildung in Pixeln)
XLabelAnzahl –> 7 (Anzahl der Beschriftungspunkte auf der X–Achse)

Zeitreihenfunktionen II - TSDetrend

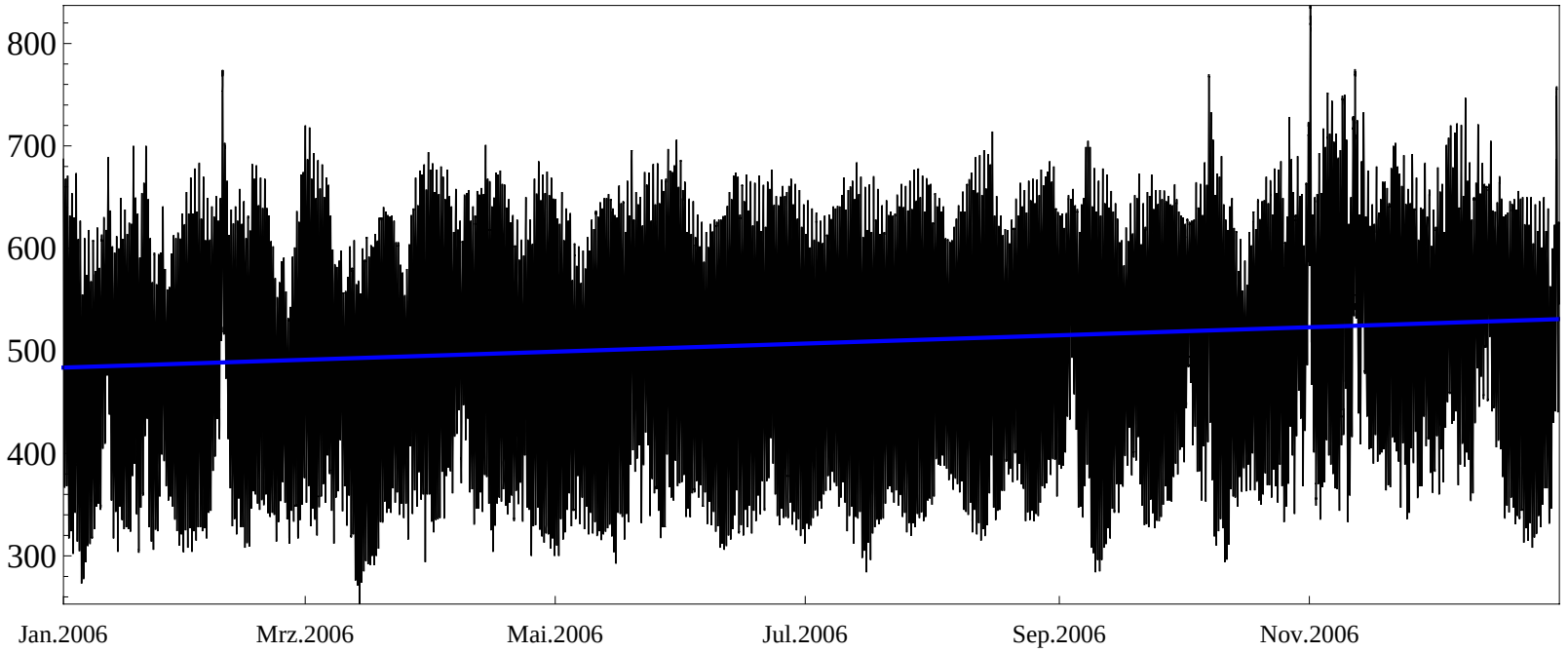
In[59]:=
```
?TSDetrend
{detrendedTS, bestfit, Mittel, Stabw, min, max} = TSDetrend[TSlong];
TSFrame[TSlong]
{bestfit, Mittel, Stabw, min, max}
Show[TSListPlotArray[TSlong][[1]], Plot[bestfit[[2]] * x + bestfit[[1]],
  {x, TSlong[[1, 1]], TSlong[[-1, 1]]}, PlotStyle → {Blue, Thick}], ImageSize → 800]
```

TSDetrend[TS] –> {detrendedTS, {b(BestFit), m(BestFit)}, Mean, Standardabweichung, Min, Max}
Die Funktion ermittelt den linearen Trend und entfernt ihn aus der Zeitreihe. Die trendbereinigte Zeitreihe, die Parameter der Trendfunktion
(Steigung m und Konstante b) sowie Mittelwert, Standardabweichung, Minimum und Maximum der trendbereinigten Zeitreihe werden zurückgegeben.

Out[61]= 01.01.2006 00:01:00 – 31.12.2006 23:51:00 –-> 52560 Datensätze. Länge: 0a364d23h50m0s

Out[62]= $\left\{\left\{-4533.31,\ 1.49984\times10^{-6}\right\},\ -1.46945\times10^{-12},\ 107.484,\ -240.103,\ 313.837\right\}$

Out[63]=


800
700
600
500
400
300
Jan.2006
Mrz.2006
Mai.2006
Jul.2006
Sep.2006
Nov.2006

◄ | ►

Zeitreihenfunktionen II - TSInterpol

In[64]:=
```
?TSInterpol
tmpTS = Take[TSlong, 100];
iTS = TSInterpol[tmpTS, tmpTS[[1, 1]], tmpTS[[-1, 1]], 5*60];
TSFrame[tmpTS]
TSFrame[iTS]
TSListPlotArray[{tmpTS, tmpTS, iTS},
  JoinedList → {True, False, False}, PointLineThickness → {0.001, 0.005, 0.003},
  ColorList → {Black, Red, Blue}, Legend → False, ISize → 800][[1]]
```

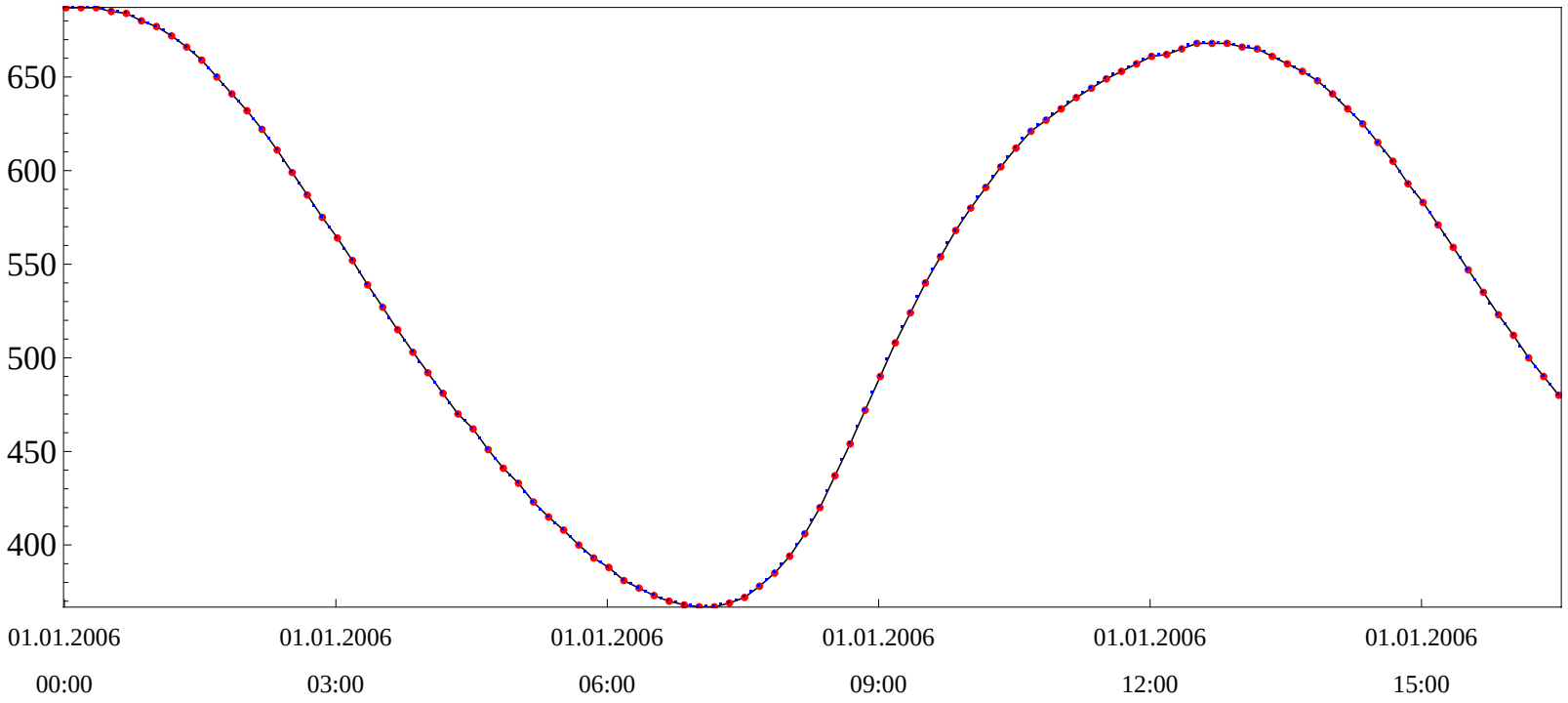
TSInterpol[TS, von, bis, dt, Optionen] –> TS
Die Funktion interpoliert aus den gegebenen Stützstellen in TS Zwischenwerte von _von_ bis _bis_ mit dem Inkrement dt in Sekunden. Neben der Zeit wir nur die zweite Spalte interpoliert; weitere Spaten bleiben unberücksichtigt. _von_ und _bis_ müssen dabei innerhalb der vorgegebenen Zeitreihe liegen, sonst wird die Funktion nicht ausgeführt.

Optionen:
Method –> Automatic (s.ListInterpolation [Spline oder Hermite])
Order –> 3 (s.ListInterpolation)
VerboseMode –> False (True –> mehr Infos)

Out[67]= 01.01.2006 00:01:00 - 01.01.2006 16:31:00 --> 100 Datensätze. Länge: 0a0d16h30m0s

Out[68]= 01.01.2006 00:01:00 - 01.01.2006 16:31:00 --> 199 Datensätze. Länge: 0a0d16h30m0s

Out[69]=


650
600
550
500
450
400
01.01.2006
00:00
01.01.2006
03:00
01.01.2006
06:00
01.01.2006
09:00
01.01.2006
12:00
01.01.2006
15:00

◄ | ►

Scheitelwertzeitreihen - Scheitelwertbestimmung

In[70]:=
```
?TSFindFirstExtrema
?TSScheitel3
?TSScheitelFull
TSFrame[TSlong]
{Tnw, Thw} = TSScheitelFull[TSlong]; // AbsoluteTiming
TSListPlotArray[{TSlong, Tnw, Thw},
  PLabel → "Zeitreihe mit Scheitelwerten", ColorList → {Black, Green, Red},
  JoinedList → {True, False, False}, PointLineThickness → {0.001, 0.005, 0.005},
  LegendEntryList → {"Zeitreihe", "Tnw", "Thw"}, ISize → 800][[1]]
```

TSFindFirstExtrema[TS, Optionen] –> {True(=Tnw)/False(=Thw), Zeit}
Die Funktion liefert den Zeitpunkt und die Art (Tnw/Thw) des ersten Extremas einer Zeitreihe TS.
Unterfunktion von TSScheitel3.

Optionen:
VerboseMode –> False (True –> mehr Infos)

TSScheitel3[TS, Optionen] –> {ListTnw, ListThw}.
Die Funktion liefert die Extrema einer Zeitreihe TS in separaten Zeitreihen für Tnw/Thw zurück.
Achtung!: Das gegebene Zeitraster bleibt erhalten. Eine Interpolation und ein Zerteilen in kleinere Abschnitte für eine höhere Genauigkeit/Performance erledigt TSScheitelFull.
Unterfunktion von TSScheitelFull.

Optionen:
LastExtrema –> unknown / [{True[=Tnw] oder False[=Thw], Zeitpunkt letztes Extrema}] (Bei unknown wird TSFindFirstExtrema benutzt, ansonsten
    wird das vorige Extrema als Anhaltspunkt für das nächste Extrema genutzt. Dabei MUSS die Zeitreihe ab dem letzten Extrema fortgesetzt werden.)
VerboseMode –> False (True –> mehr Infos)

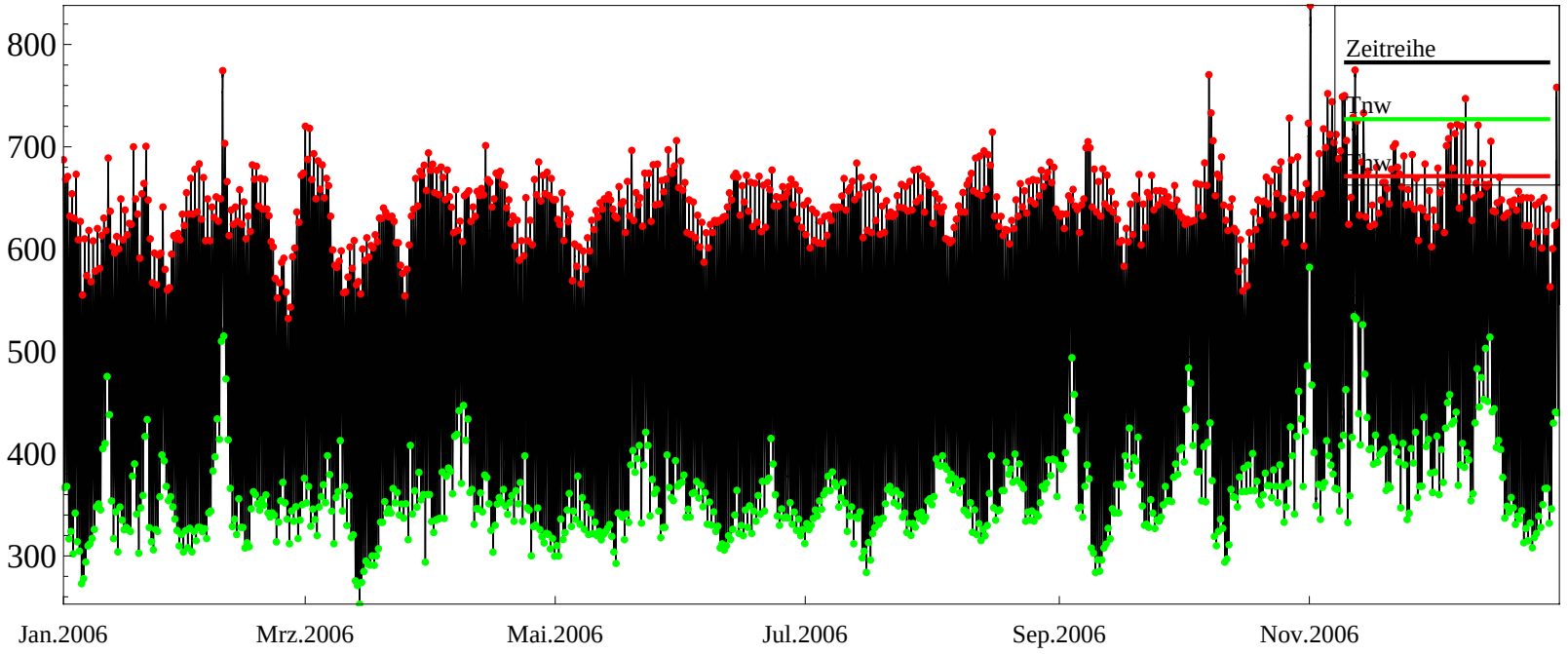
TSScheitelFull[TS, Optionen] –> {ListTnw, ListThw}.
Die Funktion liefert die Extrema einer Zeitreihe TS in separaten Zeitreihen für Tnw/Thw zurück. Es erfolgt eine
    Aufteilung der gesamten Zeitreihe in Unterzeitreihen (Blöcken). Diese Blöcke werden, wenn noch nicht geschehen, auf 1Min–Werte interpoliert
    und an die Funktion TSScheitel3 weitergereicht. Anschließend werden alle Extrema zusammengefügt und letztendlich zurückgegeben.

Optionen:
Blocklaenge –> 365*24*60*60 Sek.
MethodofInterpolation –> Spline [Spline, Hermite; s.TSInterpol]
VerboseMode –> False (True –> Zwischeninformationen für jeden Block)
DebugMode –> False (True –> sehr detaillierte Informationen mit Plots. Besser vorher die Blocklänge auf einige Tage setzen!)

Out[73]= 01.01.2006 00:01:00 – 31.12.2006 23:51:00 –> 52560 Datensätze. Länge: 0a364d23h50m0s

Out[74]= {3.5311144, Null}

Out[75]=


Zeitreihe mit Scheitelwerten
800
700
600
500
400
300
Jan.2006
Mrz.2006
Mai.2006
Jul.2006
Sep.2006
Nov.2006
Zeitreihe
Tnw
Tnw

◄ | ►

Scheitelwertzeitreihen - TScalcTidehub

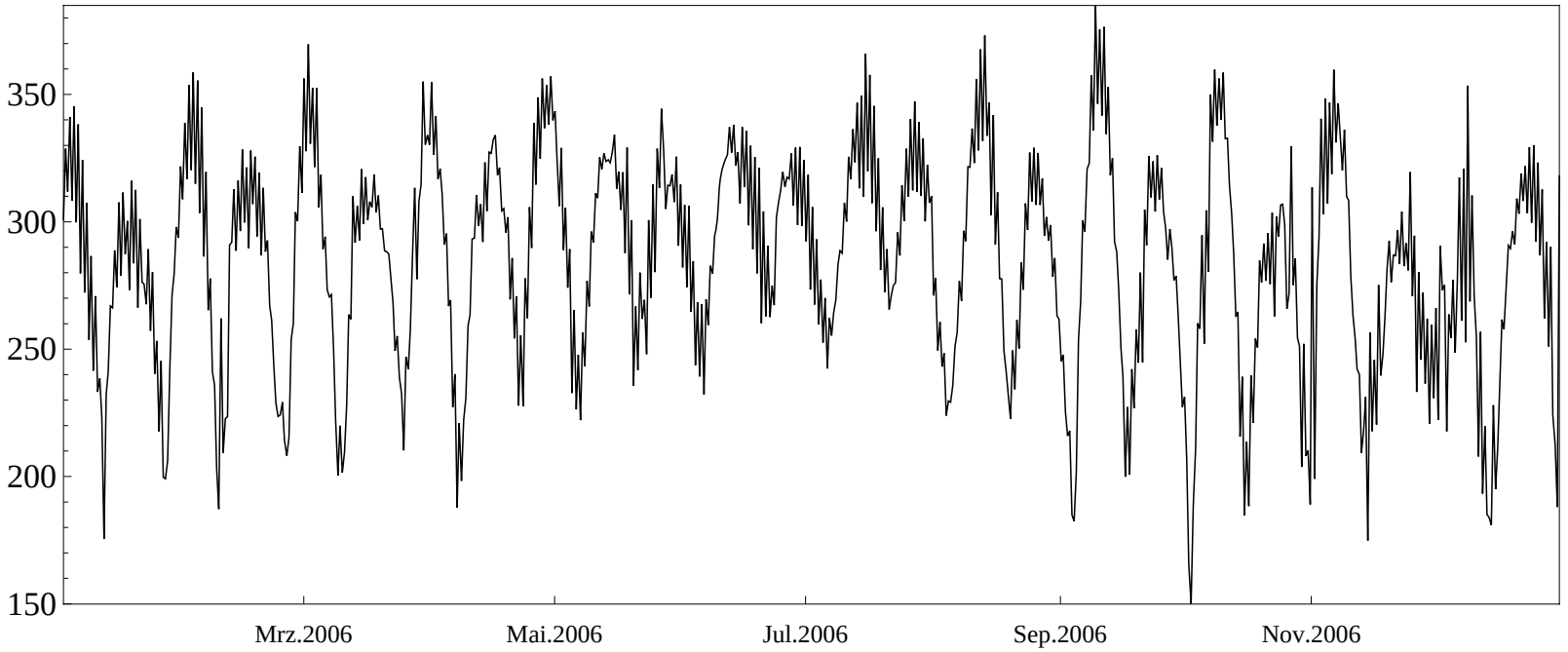
In[76]:=
```
?TScalcTidehub
TSFrame[Tnw]
TSFrame[Thw]
{Tnw, Thw, Thb} = TScalcTidehub[Tnw, Thw];
TSFrame[Tnw]
TSFrame[Thw]
TSFrame[Thb]
TSListPlotArray[Thb, ISize → 800][[1]]
```

TScalcTidehub[TnwTS, ThwTS] –> {TnwTS, ThwTS, ThbTS}
Die Funktion schneidet ggf. die Tnw– und Thw–Liste passend(überzählige Tnw/Thw werden entfernt) und berechnet
aus den beschnittenen Listen den Tidehub nach DIN4049. Die ggf. beschnittenen Listen von Tnw und Thw und die Thb–Liste werden
zurückgegeben. Die Tnw–Liste enthält ein Element mehr als die Thw/Thb–Liste. Die Zeitpunkte der Thb–Liste entsprechen denen der Thw–Liste.

Optionen:
DebugMode –> False (True –> mehr Infos)

```
Out[77]= 01.01.2006 07:07:00 – 31.12.2006 15:56:00 --> 705 Datensätze. Länge: 0a364d8h49m0s
Out[78]= 01.01.2006 00:17:00 – 31.12.2006 21:13:00 --> 706 Datensätze. Länge: 0a364d20h56m0s
Out[80]= 01.01.2006 07:07:00 – 31.12.2006 15:56:00 --> 705 Datensätze. Länge: 0a364d8h49m0s
Out[81]= 01.01.2006 12:34:00 – 31.12.2006 08:09:00 --> 704 Datensätze. Länge: 0a363d19h35m0s
Out[82]= 01.01.2006 12:34:00 – 31.12.2006 08:09:00 --> 704 Datensätze. Länge: 0a363d19h35m0s
```

Out[83]=


350
300
250
200
150
Mrz.2006
Mai.2006
Jul.2006
Sep.2006
Nov.2006

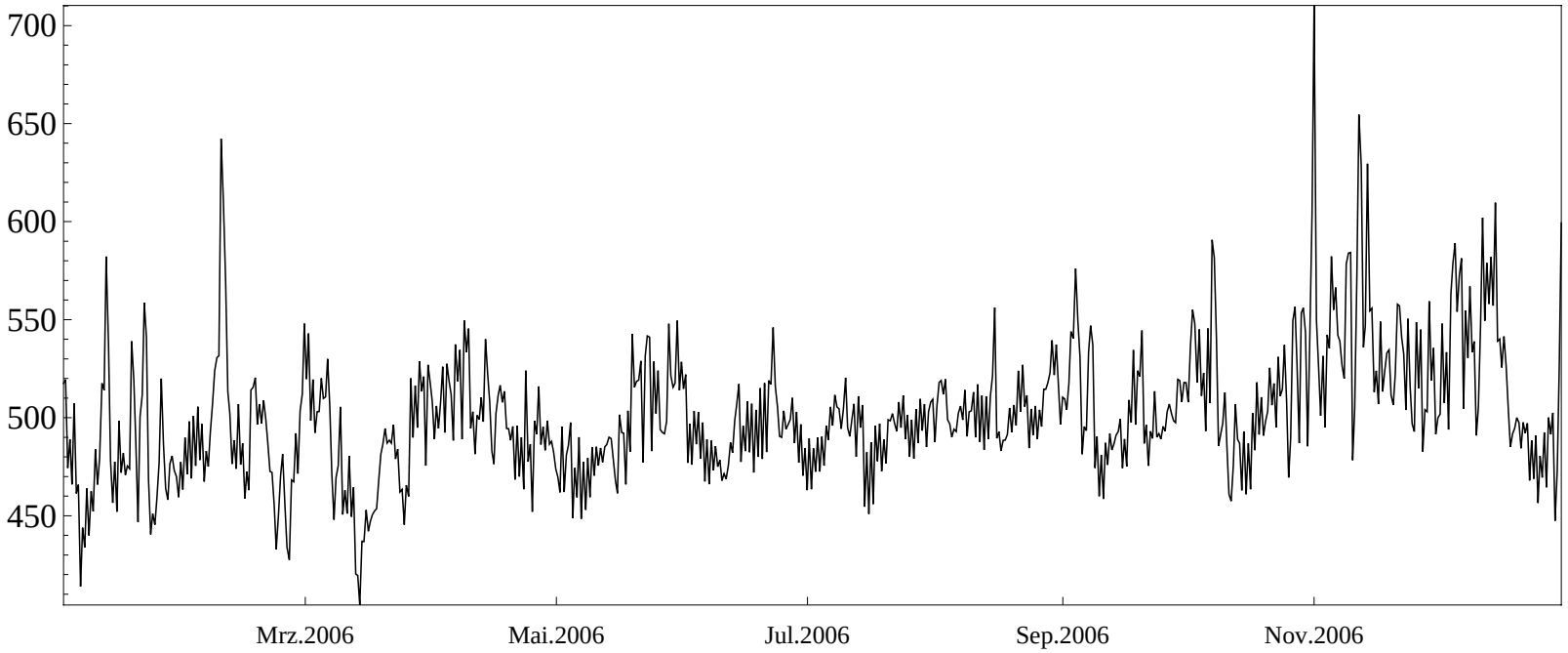
Scheitelwertzeitreihen - TScalcTidehalbwasser

In[84]:=
```
?TScalcTidehalbwasser
Thalb = TScalcTidehalbwasser[Tnw, Thw];
TSListPlotArray[Thalb, ISize → 800][[1]]
```

TScalcTidehalbwasser[TnwTS, ThwTS] –> TidehalbwasserTS
Die Funktion berechnet das Tidehalbwasser aus den Zeitreihen TnwTS und ThwTS ((Thw–Tnw)/2+Tnw).

Optionen:
DesiredTime –> Tnw; (Zeitpunkt des Tidehalbwassers [Tnw, Thw, InBetween])
VerboseMode –> False; (True –> mehr Infos)

Out[86]=


700
650
600
550
500
450
Mrz.2006
Mai.2006
Jul.2006
Sep.2006
Nov.2006

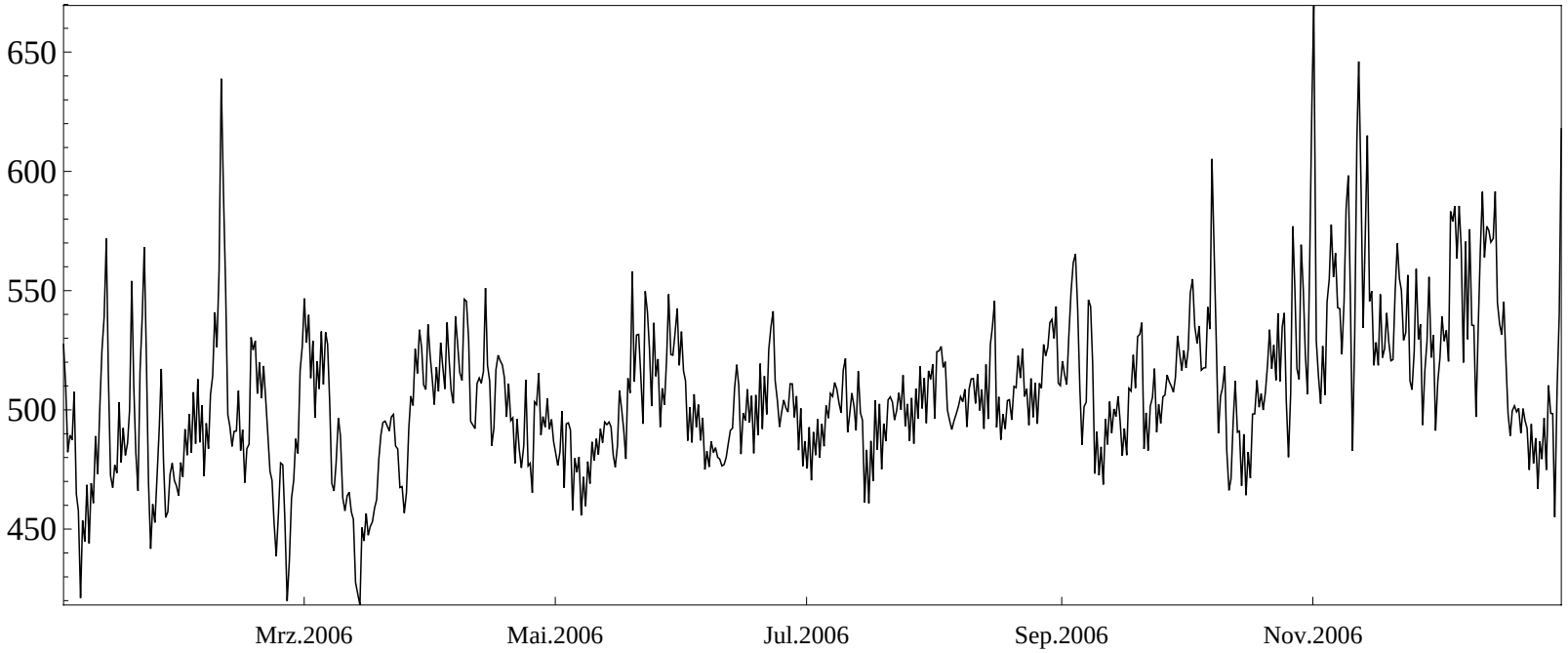
◄ | ►

Scheitelwertzeitreihen - TScalcTMW

In[87]:=
```
?TScalcTMW
Tmw = TScalcTMW[TSlong, Tnw, Thw];
TSListPlotArray[Tmw, ISize → 800][[1]]
```

TScalcTMW[TS, Tnw, Thw] –> {MittelwassernachDINTS}
Die Funktion ermittelt das arithmetische Mittel der Zeitreihe TS zwischen Tnw und Tnw+1 und legt dieses Mittel zeitlich auf das Thw (Entspricht der DIN4049).

Out[89]=


650
600
550
500
450
Mrz.2006
Mai.2006
Jul.2006
Sep.2006
Nov.2006

◄ | ►

Scheitelwertzeitreihen - TSEreignisKopplung

In[90]:=
```
?TSEreignisKopplung
{TSA, TSB} =
  TSEreignisKopplung[Thw, Tnw, TimeThreshold → {-0 * 60 * 60, 7 * 60 * 60}, VerboseMode → True];
TSFrame[TSA]
TSFrame[TSB]
```

TSEreignisKopplung[TS1, TS2 , Optionen] –> {{TSA ,TSB}}
Die Funktion verknüpft zwei Zeitreihen TS1 und TS2 miteinander und liefert zwei neue Zeitreihen TSA/TSB. Über die Option TimeThreshold wird angegeben, in welchem zeitlichen Mini–/Maximalabstand sich die Ereignisse der Zeitreihen TS1 und TS2 zueinander bewegen dürfen. Ist der Zeitabstand von TS1 zu TS2 größer oder kleiner als TimeThreshold erlaubt, wird die jeweilige Zeitreihe solange beschnitten, bis der Abstand der Anfangswerte von TS1 und TS2 unter TimeThreshold liegt. Jedes Ereignis in TS1 kann nur EINMAL einem Ereignis in TS2 zugewiesen werden. Die zurückgegebenen Zeitreihen TSA/TSB haben die gleiche Länge.

Optionen:
TimeThreshold –> {0 ,8∗60∗60) ({dtvor, dtnach}; z.B. {–1∗60∗60, 8∗60∗60} –> ein Ereignis wird einem anderen zugeordnet, wenn der zeitliche Abstand der beiden Ereignisse zwischen –1∗60∗60 und 8∗60∗60 Sekunden liegt)
VerboseMode –> False (True –> mehr Info)
DebugMode –> False (True –> Zusatzinfo)
ReturnProblems –> False (True –> gibt zusätzlich eine Liste der Kopplungszeitpunkte zurück, die überprungen wurden [{{TS1a,TS2a}, ...}])

```
Anzahl der Paare: 649
Anzahl der übersprungenen A: 55
Anzahl der übersprungenen B: 54
Mittleres dt [sec]/[min]: 23526.5 / 392.108
StandardAbweichung: 942.25
Min dt: 17820
Max dt: 25200
```

Out[92]= 01.01.2006 12:34:00 – 30.12.2006 20:09:00 –-> 649 Datensätze. Länge: 0a363d7h35m0s

Out[93]= 01.01.2006 19:26:00 – 31.12.2006 01:06:00 –-> 649 Datensätze. Länge: 0a363d5h40m0s

Zeitreihenfunktionen II - TScalcTimeshift

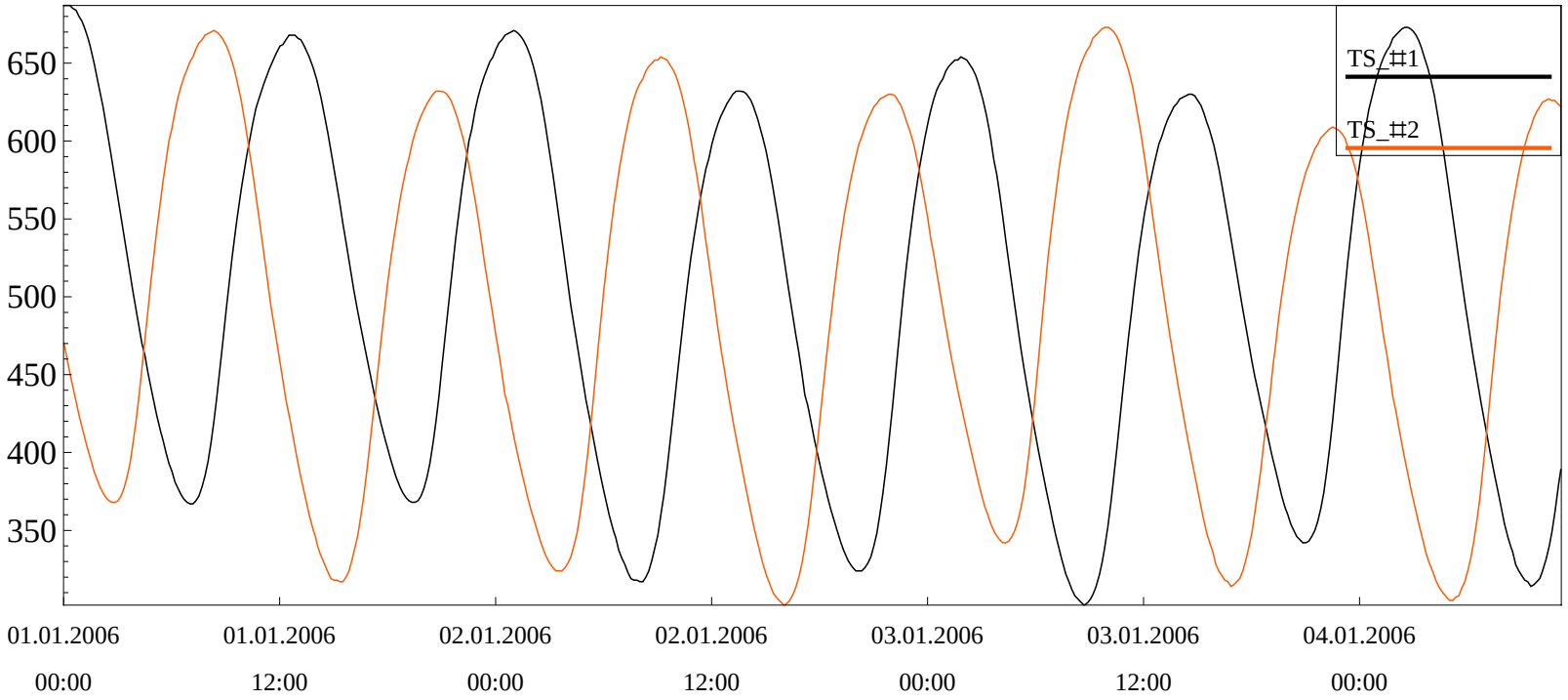
In[94]:=
```
?TScalcTimeshift
tmpTSA = Take[TSlong, {1, 500}];
tmpTSB = Transpose[{tmpTSA[[All, 1]], Take[TSlong, {101, 600}][[All, 2]]}];
TSListPlotArray[{tmpTSA, tmpTSB}, ISize → 800][[1]]
TScalcTimeshift[tmpTSA, tmpTSB, VerboseMode → True]
```

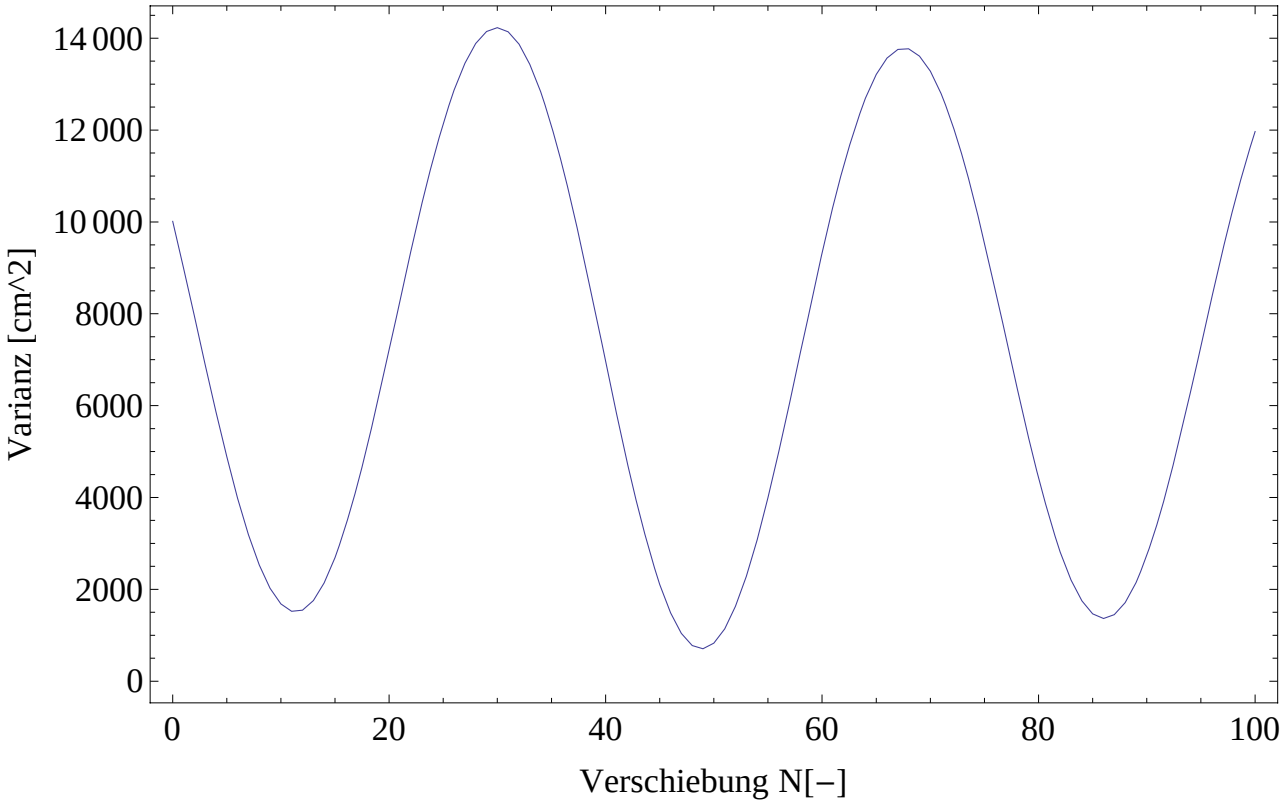
TScalcTimeshift[TS1, TS2, Optionen] –> Gruppenlaufzeit in Minuten
Bedingungen:
Die Zeitreihe TS1 MUSS zeitlich vor der Zeitreihe TS2 liegen (Maxima in TS1 kommt vor Maxima TS2).

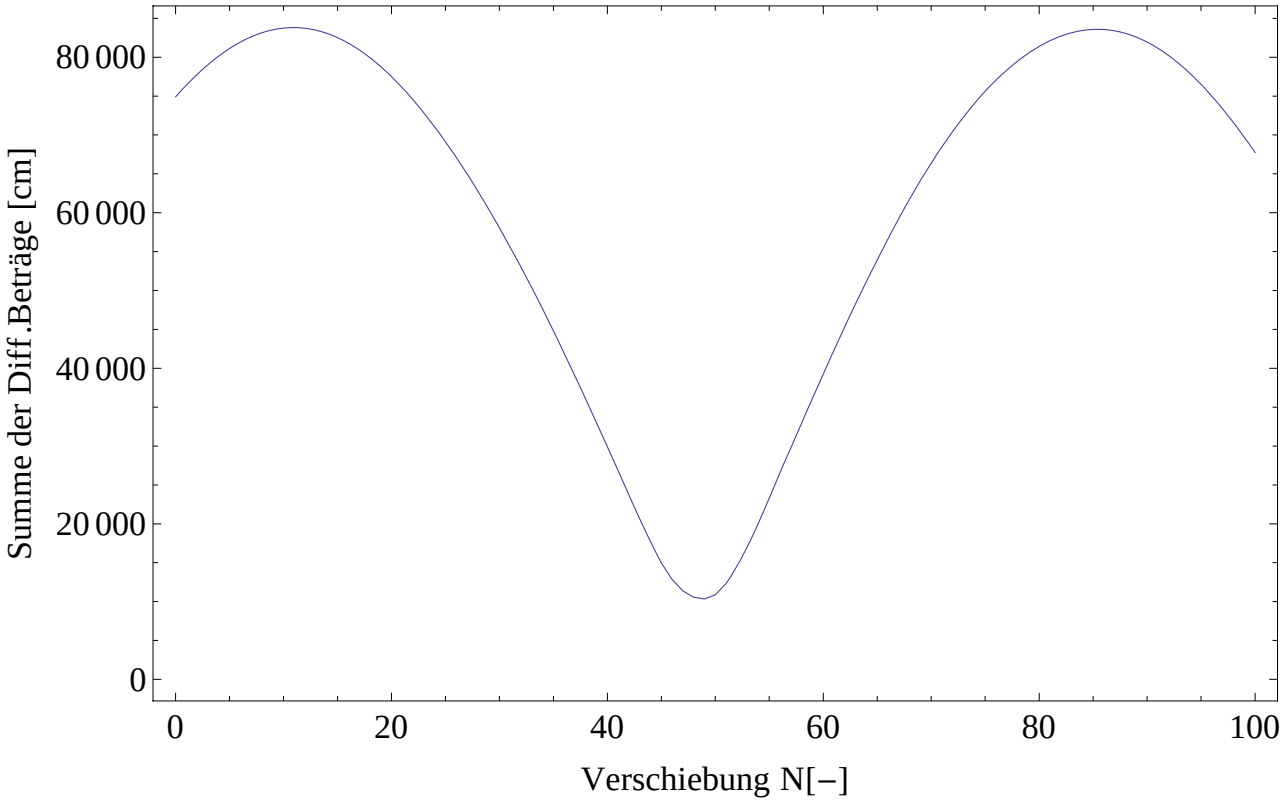
Die Funktion extrahiert zwei Sub–Zeitreihen mit der Länge Min[Length[TS1],Length[TS2]–MaxN]: die erste aus TS1 ab dem Datenpunkt
1 bis <Länge>, die zweite aus TS2 ab dem Datenpunkt i=0...MaxN bis <Länge>. Über Methode wird selbige zur Bestimmung ausgewählt.
LinearModel: Bei einer Regression zwischen den Zeitreihen mit einem linearen Modell wird die Schätzvarianz bestimmt.
Distance: die Summe der Differenzen wird berechnet.
Im Anschluß wird das Minimum (Schätzvarianz oder Summe der Differenzen) bestimmt. Die zeitliche Verschiebung, bei der das Minimum auftritt wird zurückgegeben.

Optionen:
MaxN –> 100 (Verschiebung von 0 bis MaxN)
Method –> Both (LinearModel, Distance)
ResampleMinute –> False (True –> TS1 und TS2 werden auf Minutenabstand interpoliert)
VerboseMode –> False (True –> Diagramm[Zerschiebung zu Schätzvarianz/Summe der Differenzen])
DebugMode –> False (True –> DebugInformationen)

Out[97]=


650
600
550
500
450
400
350
TS_#1
TS_#2
01.01.2006
00:00
01.01.2006
12:00
02.01.2006
00:00
02.01.2006
12:00
03.01.2006
00:00
03.01.2006
12:00
04.01.2006
00:00


14 000
12 000
10 000
8000
6000
4000
2000
0
Varianz [cm^2]
0
20
40
60
80
100
Verschiebung N[−]




```
minimaler Versatz Regression bei n = 49
Minimum der Summe der Beträge der Differenzen bei n = 49
```

Out[98]= 490

Zeitreihenfunktionen II - TScalcMean / TScalcMeanFull

In[99]:=
```
?TScalcMean
?TScalcMeanFull
TSFrame[TSlong]
TSFrame[Tnw]
MTK = TScalcMeanFull[TSlong, Tnw, 100, MinTimeLength → 630 * 60, VerboseMode → True];
TSListPlotArray[MTK, ISize → 800][[1]]
```

TScalcMean[ArrayTS, n, Optionen] –> {Mittelwert1, Mittelwert2, ... Mittelwertn}
Die Funktion gibt die mittlere Zeitreihe zurück. Dabei muss ArrayTS in der Form {TS1, TS2, ... TSN} sein. Nur die zweite Spalte der Zeitreihen wird gemittelt! Die Zahl n gibt die Anzahl der Stützstellen an, auf die die einzelnen Zeitreihen mit einem kubischen Spline umgesampelt werden.

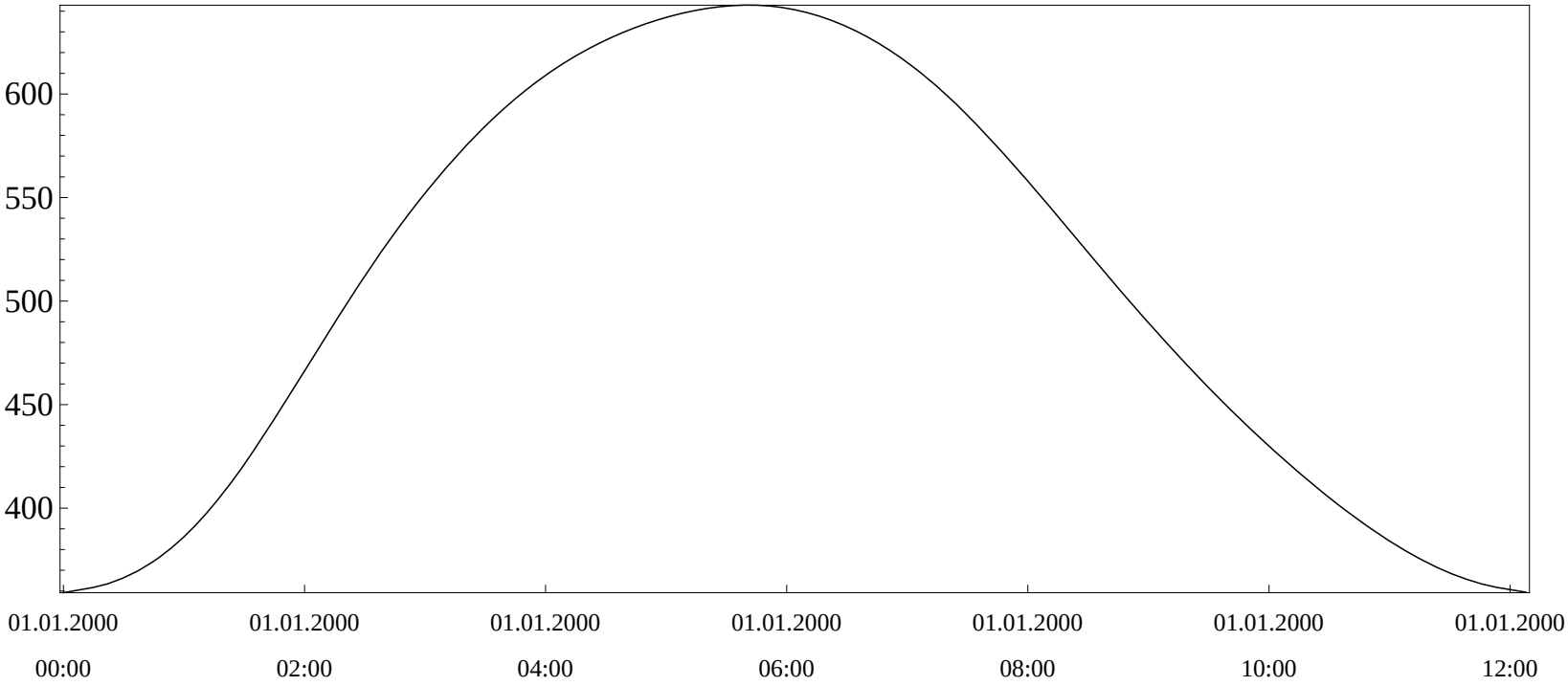
Optionen:
VerboseMode –> False (True –> mehr Infos)

TScalcMeanFull[TS1, TS2, n, Optionen] –> TS
Die Funktion zerschneidet die Funktion TS1 an den gegebenen Zeitpunkten TS2 (z.B. Tnw) ([Zeitpunkt1,Zeitpunkt2[, [Zeitpunkt2,Zeitpunkt3[, ...) und ermittelt für jede Spalte von TS eine mittlere Zeitreihe über TScalcMean. n ist dabei die Anzahl der Stützstellen, die TScalcMean zum interpolieren benötigt. Die zurückgegebene Zeitreihe beginnt ab der Zeit StartTime und endet nach der mittleren Länge aller zur Mittelung verwendeten Zeitreihen.

Optionen:
MinTimeLength –> 700*60 (in Sekunden; die zerschnittenen Zeitreihen müssen mindestens diese Dauer aufweisen, damit sie zur Mittelung verwendet werden. Ansonsten werden sie ignoriert. Hinweis: gerade am Anfang und Ende der Zeitreihe kann es vorkommen, dass unvollständige Tiden vorkommen, die können über dieses Kriterium ausgeschlossen werden.)
StartTime–>AbsoluteTime[{2000,1,1,0,0,0}] (Vorgabe für den Beginn der mittleren Zeitreihe die zurückgegeben wird)
VerboseMode –> False (True –> mehr Infos)

```
Out[101]= 01.01.2006 00:01:00 – 31.12.2006 23:51:00 --> 52560 Datensätze. Länge: 0a364d23h50m0s

Out[102]= 01.01.2006 07:07:00 – 31.12.2006 15:56:00 --> 705 Datensätze. Länge: 0a364d8h49m0s

Anzahl der Zeitreihen: 704
Kürzeste Zeitreihe [min]: 640.
Längste Zeitreihe [min]: 880.
Mittlere Länge [min]: 735.298
```

Out[104]=



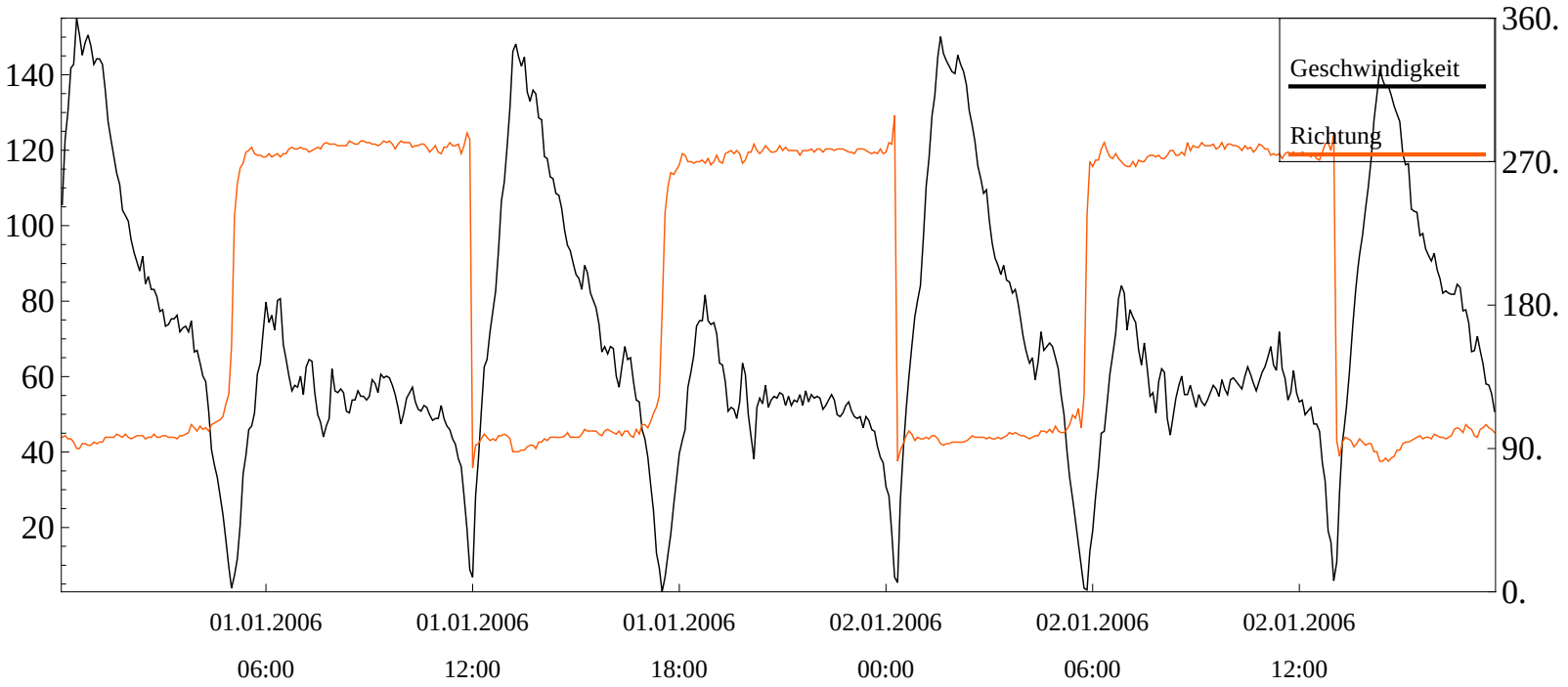
Strömungszeitreihen - TScuDetectMainDirections

In[105]:=
```
?TScuDetectMainDirections
cuTS = Get["D:\\Analyse\\_Library\\TestCU.mm"];
cuTS2 = Get["D:\\Analyse\\_Library\\TestCU2.mm"];
TSListPlotArray[TSMultiToArray[cuTS2], SecondAxisList → {False, True},
  LegendEntryList → {"Geschwindigkeit", "Richtung"}, YRange2 → {0, 360}, ISize → 800][[1]]
TScuDetectMainDirections[cuTS2, VerboseMode → True]
```

TScuDetectMainDirections[TScu, Optionen] –> {Maximum im Richtungsintervall [0–180[Grad, Maximum im Richtungsintervall [180–360[Grad}
Die Funktion bestimmt aus einer Strömungszeitreihe TScu der Form {{Zeit1, Wert1, Richtung1},{Zeit2, Wert2, Richtung2},...} die Hauptrichtung über das Maximum der Richtungsverteilung in Ein–Grad–Schritten. Die zweite Hauptrichtung ist um 180 Grad zur ersten Hauptrichtung versetzt. Sind mehrere Maxima gleicher Höhe in der Richtungsverteilung vorhanden, wird eine Fehlermeldung und ein Plot der Richtungsverteilung ausgegeben und das erste Maximum wird als das Hauptmaximm interpretiert.

Optionen:
VerboseMode –> False (True –> mehr Infos)

Out[108]=




```
Hauptrichtung : 97
erwartetet zweite Hauptrichtung : 277
```

Out[109]= {97, 277}

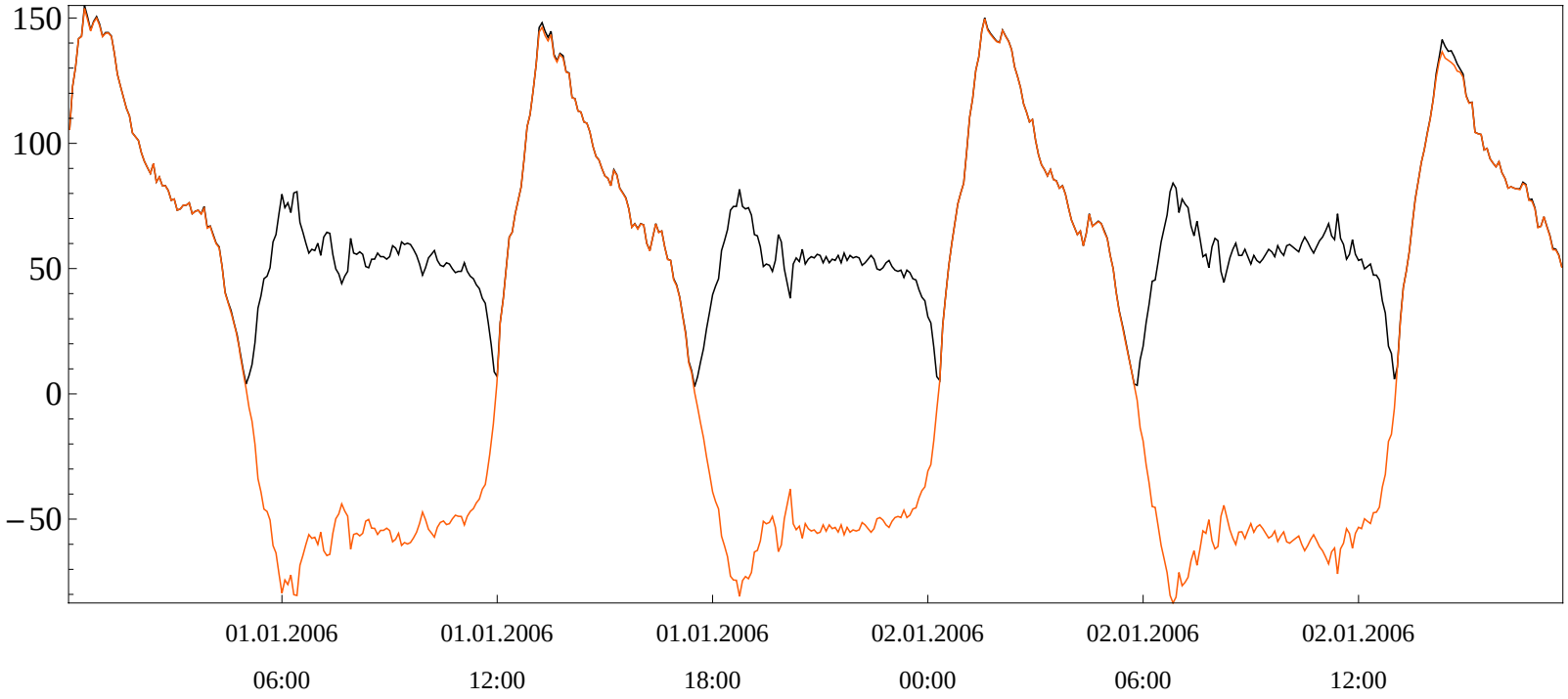
Strömungszeitreihen - TScuNormalize

In[110]:=
```
?TScuNormalize
nTS = TScuNormalize[cuTS2, 97];
TSListPlotArray[{cuTS2, nTS}, ISize → 800][[1]]
```

TScuNormalize[TScu_, nord_] –> TS

Die Funktion normalisiert eine Strömungszeitreihe TScu {{Zeit1, Wert1, Richtung1},{Zeit2, Wert2, Richtung2},...} auf die Hauptrichtung (nord). Aus einer betraglichen Strömung mit Richtungsangabe wird eine Strömung mit alternierendem Vorzeichen über die Nordachse.

Out[112]=


150
100
50
0
−50
01.01.2006
06:00
01.01.2006
12:00
01.01.2006
18:00
02.01.2006
00:00
02.01.2006
06:00
02.01.2006
12:00

◄ | ►

Strömungszeitreihen - TScucalcKenterpunkte

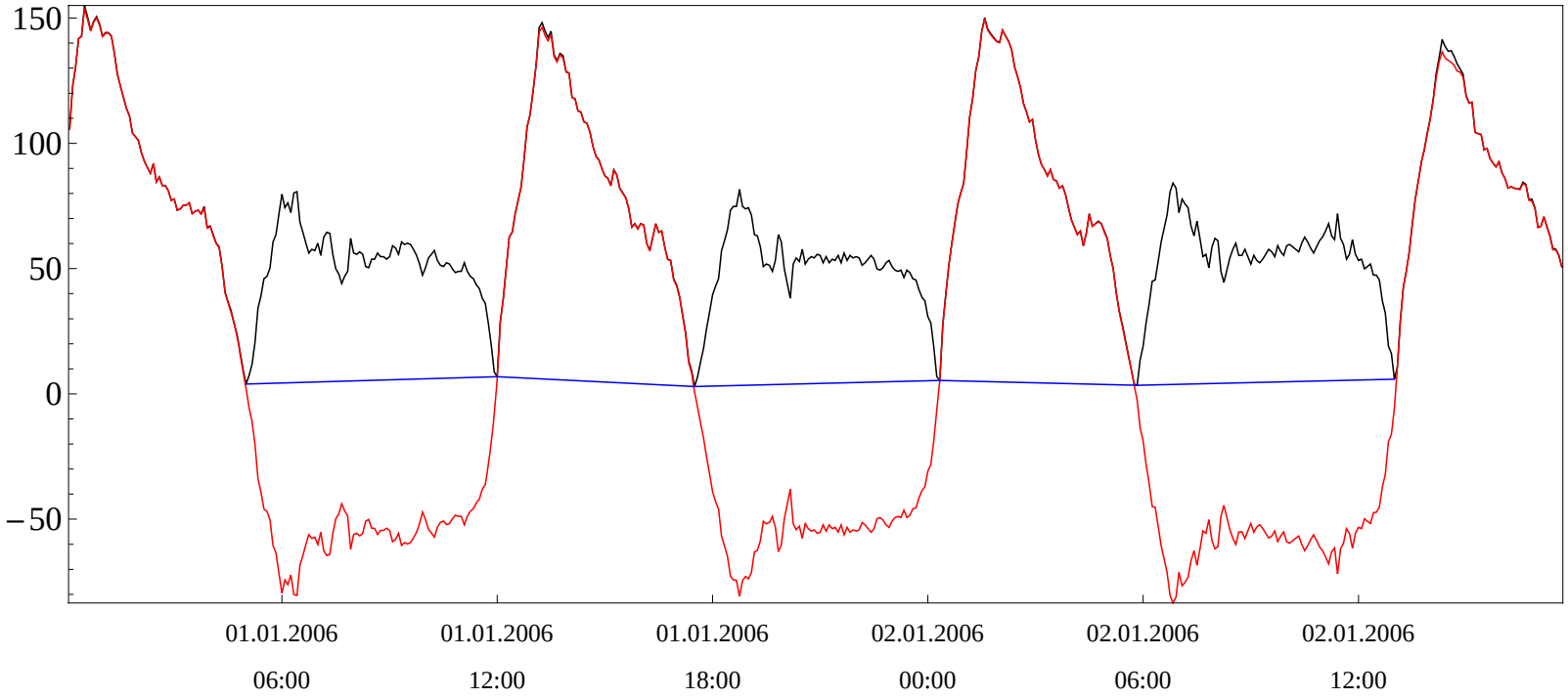
In[113]:=
```
?TScucalcKenterpunkte
kpTS = TScucalcKenterpunkte[cuTS2, 1 * 60 * 60, MainDirection → Automatic]
nTS = TScuNormalize[cuTS2, TScuDetectMainDirections[cuTS2][[1]]];
TSListPlotArray[{cuTS2, nTS, kpTS},
  ColorList → {Black, Red, Blue}, Legend → False, ISize → 800][[1]]
```

TScucalcKenterpunkte[TS, minTimebetweenKP, Optionen] –> KenterpunkteTS
Die Funktion bestimmt die Kenterpunkte von TS {{Zeit1, Wert1, Richtung1},{Zeit2, Wert2, Richtung2},...}.
Steht die Option MainDirection auf Automatic wird zuerst über TScuDetectMainDirections die Hauptrichtung bestimmt. Ansonsten muss die Nordrichtung über MainDirection gesetzt werden. Anschließend wird über die Funktion TScuNormalize, die Zeitreihe normalisiert. Es folgt die Bestimmung des Nulldurchgangs (über den Vorzeichenwechsel). Der Zeitpunkt des Nulldurchgangs (auf dem gegebenen Zeitraster) wird zum Kenterpunkt bestimmt. Der Wert der Strömung wird in der zweiten Spalte mit ausgegeben.
Da beim Nulldurchgang die Richtung oszillieren kann, wird anschließend auf einen zeitlichen Mindestabstand (minTimebetweenKP [Sekunden]) der Kenterpunkte geprüft. Kenterpunkte, die einen kleineren Zeitabstand als vorgegeben (minTimebetweenKP [Sekunden]) haben, werden aussortiert und es wird eine Warnung generiert (ausgewählt wird immer der erste Kenterpunkt, die folgenden werden ggf. aussortiert).

Optionen:
MainDirection –> Automatic (Automatic –> es wird die Hauptrichtung über TScuDetectMainDirections bestimmt, sonst Wert übergeben)
VerboseMode –> False (True –> mehr Infos)

Out[114]=
```
{{3 345 080 400, 3.91, 0}, {3 345 105 600, 6.84, 1}, {3 345 125 400, 2.93, 0},
 {3 345 150 000, 5.38, 1}, {3 345 169 800, 3.42, 0}, {3 345 195 600, 5.87, 1}}
```

Out[116]=


150
100
50
0
−50
01.01.2006
06:00
01.01.2006
12:00
01.01.2006
18:00
02.01.2006
00:00
02.01.2006
06:00
02.01.2006
12:00

◄ | ►

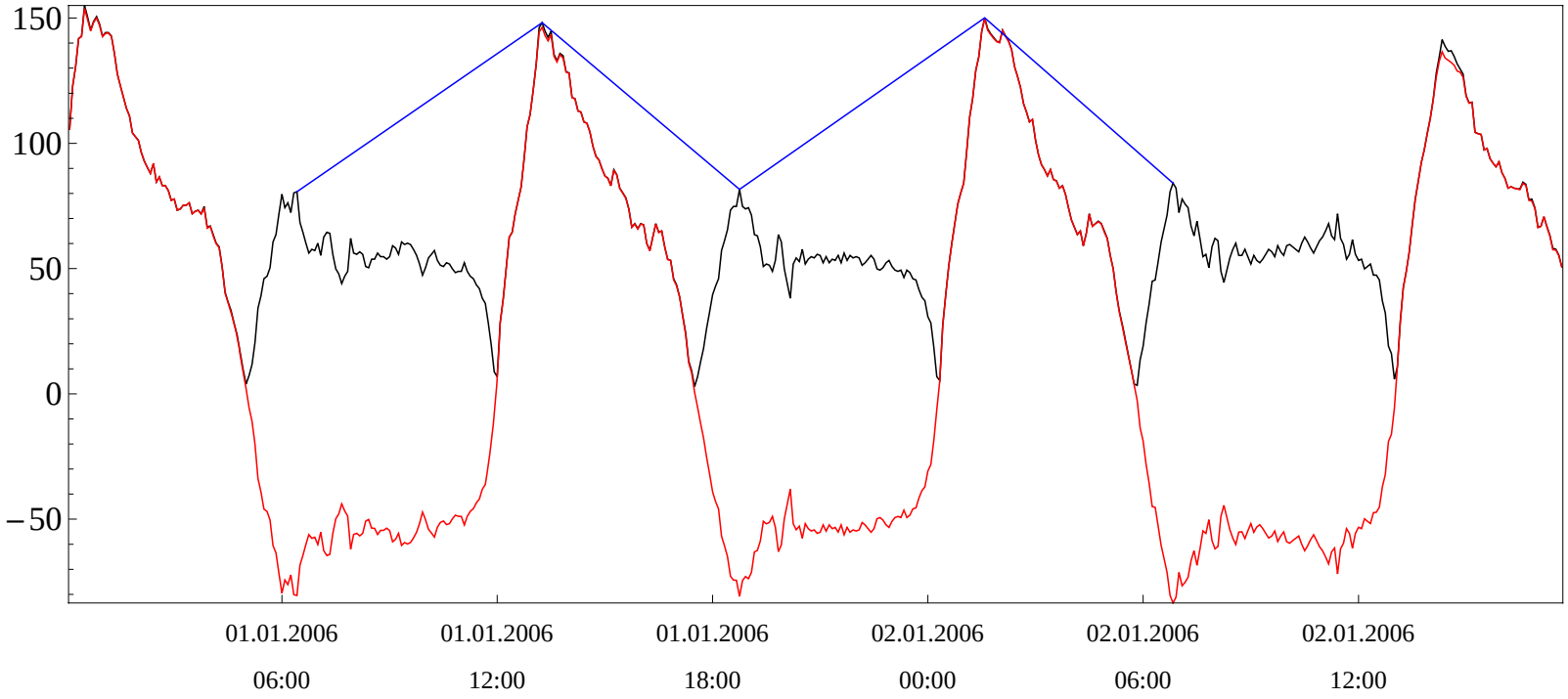
Strömungszeitreihen - TScucalcVMax

In[117]:=
```
?TScucalcVMax
vmaxTS = TScucalcVMax[cuTS2, kpTS]
TSListPlotArray[{cuTS2, nTS, vmaxTS},
  ColorList → {Black, Red, Blue}, Legend → False, ISize → 800][[1]]
```

TScucalcVMax[TScu, Zeit−/Kenterpunktliste{{KP1, Wert1}, {KP2, Wert2},...}] −> VMaxListe {{VMax−Zeit1, Wert1}, {VMax−Zeit2, Wert2}, ...}
Die Funktion ermittelt das Maxima einer Strömungszeitreihe TScu {{Zeit1, Wert1, Richtung1},{Zeit2, Wert2, Richtung2},...} zwischen zwei aufeinanderfolgenden Kenterpunkten und gibt diese als neue Zeitreihe aus. Sollten mehrere gleichgroße Maxima zwischen zwei Kenterpunkten vorliegen, wird das erste Maximum ausgewählt.

Out[118]=
```
{{3 345 085 500, 80.67, 273, 0}, {3 345 110 100, 148.14, 88, 1},
 {3 345 129 900, 81.65, 269, 0}, {3 345 154 500, 150.09, 93, 1}, {3 345 173 400, 84.09, 270, 0}}
```

Out[119]=


150
100
50
0
−50
01.01.2006
06:00
01.01.2006
12:00
01.01.2006
18:00
02.01.2006
00:00
02.01.2006
06:00
02.01.2006
12:00

◄ | ►

Filter - TSMovingWindow

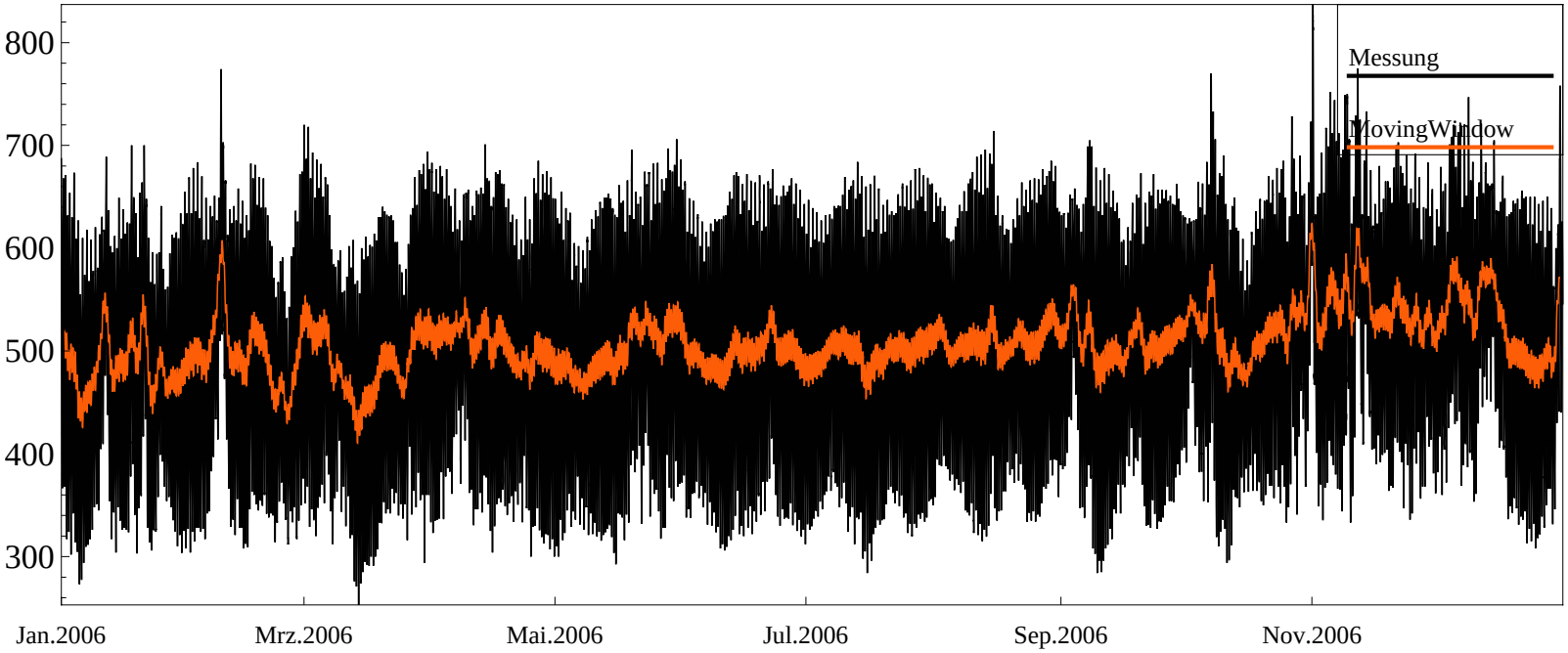
In[120]:=
```
?TSMovingWindow
TSFrame[TSlong]
fTS = TSMovingWindow[TSlong, 250]; // AbsoluteTiming
TSListPlotArray[{TSlong, fTS},
  LegendEntryList → {"Messung", "MovingWindow"}, ISize → 800][[1]]
```

TSMovingWindow[TS, n] –> {TS}
Die Funktion berechnet das arithmetische Mittel über n–Samples aus der zweiten Spalte der Zeitreihe. Verschiebung
um Eins –> Mittelwert über n–Samples –> Verschiebung um Eins, ... Der zugehörige Zeitpunkt wird berechnet durch: ((Tend–Tanf)/2)+Tanf.

Out[121]= `01.01.2006 00:01:00 – 31.12.2006 23:51:00 –-> 52560 Datensätze. Länge: 0a364d23h50m0s`

Out[122]= `{3.0155092, Null}`

Out[123]=


800
700
600
500
400
300
Messung
MovingWindow
Jan.2006
Mrz.2006
Mai.2006
Jul.2006
Sep.2006
Nov.2006

◄ | ►

Filter - TSLowPassFilter

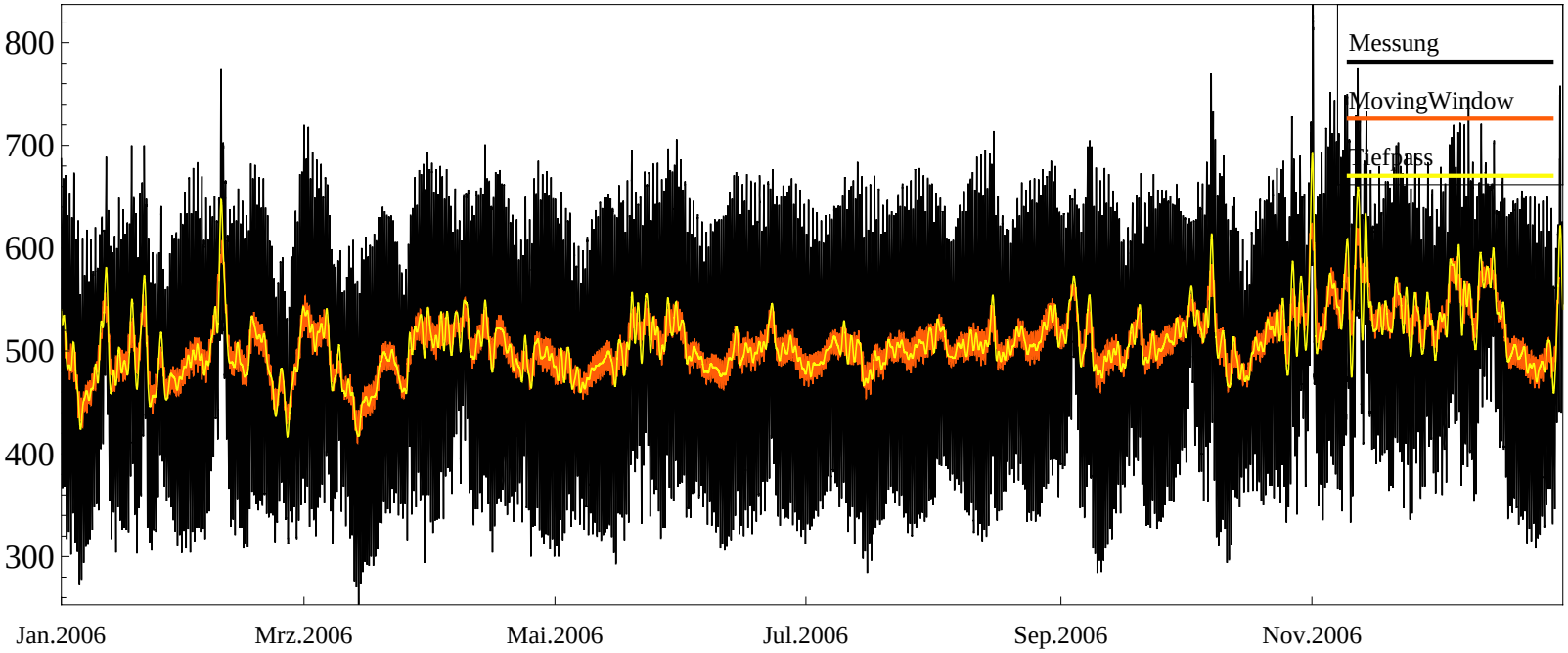
In[124]:=
```
?TSLowPassFilter
fTS2 = TSLowPassFilter[TSlong, 13]; // AbsoluteTiming
TSListPlotArray[{TSlong, fTS, fTS2},
  LegendEntryList → {"Messung", "MovingWindow", "Tiefpass"}, ISize → 800][[1]]
```

TSLowPassFilter[TS, Grenzfrequenz[Grad/h], Optionen] –> TS
Die Funktion ist ein Butterworth–Filter N–ter Ordnung, der im Fourierraum angewendet wird.

Optionen:
Order –> 10 (gibt die Ordnung N des Filters an)
Detrend –> False (True –> die Zeitreihe TS wird vor der Filterung mit TSDetrend
vom Trend befreit; nach der Filterung wird der Trend wieder hinzugefügt ––> ggf.Verminderung von Gibbs–Effekten)
ShowInfo –> False (True –> zeigt den Filter im Frequenzraum)
Epsilon –> 0.01 (An diesen Punkten wird die Frequenz berechnet, an der der Filter die Amplitude 1–Epsilon bzw. 0+Epsilon hat [nur wenn ShowInfo–>True].)

Out[125]=
```
{0.4374832, Null}
```

Out[126]=


800
700
600
500
400
300
Jan.2006
Mrz.2006
Mai.2006
Jul.2006
Sep.2006
Nov.2006
Messung
MovingWindow
Tiefpass

Partialtiden - InitPTList(Q)

In[127]:=
```
?InitPTList
?InitPTListQ
ptlist = InitPTList["D:\\Pegel\\Analyse\\Tide\\Minutenwerte\\tidegrunddaten.dat",
   DefaultAmplitudeValue → 100, MKToolsMM7`VerboseMode → False, NegativList → Automatic];
TableForm[ptlist, TableHeadings → {Automatic,
   {"Name", "Art(AT/SWT)", "Frequenz [Grad/h]", "-", "-"}}]
```

InitPTList[Directory und Filename, Optionen] –> PTList = {{Name, Art(1=AstronomischeTide, 0=FlachwasserTide), Frequenz[Grad/h], DefaultAmplitudeValue, 0}, ...}
Die Funktion lädt, konvertiert und sortiert eine Partialtidenliste (z.B. tidegrunddaten.dat).

Optionen:
NegativList –> Automatic (Off –> keine Negativliste, Automatic –> folgende PT werden entfernt, falls in der geladenen Datei(Partialtidenliste) vorhanden:
2MK_2,2MN_2,2MS_2,2NM_2,ALPHA_2,K_1[sw],K_2[sw],KO_0,KO_2,KP_2,KQ_2,MK_0,MK_1,MKN_2,MKS_2,MN_0,MNK_2,MNS_2,MO_1,MP_1,MQ_1,MQ_3,MS_0,MSK_2,NK_1, NO_1[sw],OP_2,S101–3,S101–4,S101–6,S105–3,S105–4,S105–6,S111–3,S111–4,S111–6,S111–7,S116–3,S116–4,S116–6,S116–7,S29–3,S29–4,S29–6,S41–4,S41–6,S59–4, S59–6,S59–7,S61–3,S61–4,S61–6,S61–7,S63–3,S63–4,S63–6,S63–7,S74–3,S74–4,S74–6,S74–7,S76–3,S76–4,S76–6,S76–7,S78–3,S78–4,S78–6,SK_1,SO_1[sw],SO_3,SP_1, unnamed4)
DefaultAmplitudeValue –> 200 (Alt und sollte keine Bedeutung mehr haben – wird aber aus Kompatibilitätsgründen so belassen.)
VerboseMode –> False (True –> mehr Infos)

InitPTListQ[] –> PTList = {{Name 1, Art(1=AstronomischeTide, 0=FlachwasserTide), Frequenz[Grad/h] 1, 200, 0}, ...}
Die gibt eine Liste zurück wie InitPTList bestehend aus 148 Partialtiden. Im Unterschied zur
Funktion InitPTList sind alle Parameter in der Bibliothek fest vorgegeben, womit sich die Zugriffsgeschwindigkeit erheblich erhöht.

Out[130]//TableForm=

	Name	Art (AT/SWT)	Frequenz [Grad/h]	–	–
1	Sa	1	0.0410667	100	0
2	Ssa	1	0.0821373	100	0
3	MSm	1	0.471521	100	0
4	Mm	1	0.544375	100	0
5	MSf	1	1.0159	100	0
6	Mf	1	1.09803	100	0
7	MStm	1	1.56955	100	0
8	Mtm	1	1.64241	100	0
9	MSqm	1	2.11393	100	0
10	Mqm	1	2.18678	100	0
11	2Q_1	1	12.8543	100	0
12	SIGMA_1	1	12.9271	100	0
13	neu01	0	13.3483	100	0
14	Q_1	1	13.3987	100	0
15	RHO_1	1	13.4715	100	0
16	neu02	0	13.788	100	0
17	O_1	1	13.943	100	0
18	TAU_1	1	14.0252	100	0
19	M_1	1	14.4921	100	0

20	NO_1	1	14.4967	100	0
21	KAPPA_1	1	14.5695	100	0
22	PI_1	1	14.9179	100	0
23	P_1	1	14.9589	100	0
24	S_1	1	15.	100	0
25	K_1	1	15.0411	100	0
26	PSI_1	1	15.0821	100	0
27	PHI_1	1	15.1232	100	0
28	THETA_1	1	15.5126	100	0
29	J_1	1	15.5854	100	0
30	SO_1	1	16.057	100	0
31	OO_1	1	16.1391	100	0
32	NU_1	1	16.6835	100	0
33	neu04	0	25.9364	100	0
34	neu05	0	26.3258	100	0
35	neu07	0	26.4808	100	0
36	3M2K_2	0	26.788	100	0
37	neu08	0	26.8291	100	0
38	3M{SK}_2	0	26.8702	100	0
39	3M2S_2	0	26.9523	100	0

40	MVS_2	0	27.3324	100	0
41	OQ_2	0	27.3417	100	0
42	EPS_2	1	27.4238	100	0
43	neu09	0	27.4967	100	0
44	O_2	0	27.8861	100	0
45	2N_2	1	27.8954	100	0
46	MU_2	1	27.9682	100	0
47	neu11	0	28.0411	100	0
48	3M{SN}_2	0	28.3483	100	0
49	3M{KN}_2	0	28.4304	100	0
50	N_2	1	28.4397	100	0
51	NU_2	1	28.5126	100	0
52	unnamed1	1	28.902	100	0
53	GAMMA_2	1	28.9113	100	0
54	unnamed2	1	28.943	100	0
55	M_2	1	28.9841	100	0
56	unnamed3	1	29.0252	100	0
57	DELTA_2	1	29.0662	100	0
58	neu12	0	29.1391	100	0
59	LABDA_2	1	29.4556	100	0

60	neu13	0	29.4874	100	0
61	L_2	1	29.5285	100	0
62	T_2	1	29.9589	100	0
63	S_2	1	30.	100	0
64	R_2	1	30.0411	100	0
65	3M2N_2	0	30.0729	100	0
66	K_2	1	30.0821	100	0
67	MSN_2	0	30.5444	100	0
68	ZETA_2	1	30.5537	100	0
69	ETA_2	1	30.6265	100	0
70	2SM_2	0	31.0159	100	0
71	neu14	0	31.098	100	0
72	2KM_2	0	31.1802	100	0
73	NO_3	0	42.3828	100	0
74	MO_3	0	42.9271	100	0
75	M_3	1	43.4762	100	0
76	NK_3	0	43.4808	100	0
77	MP_3	0	43.943	100	0
78	MK_3	0	44.0252	100	0
79	SP_3	0	44.9589	100	0

80	SK_3	0	45.0411	100	0
81	K_3	0	45.1232	100	0
82	neu15	0	55.8543	100	0
83	neu16	0	55.9364	100	0
84	2MNS_4	0	56.4079	100	0
85	neu17	0	56.4808	100	0
86	3MK_4	0	56.8702	100	0
87	3MS_4	0	56.9523	100	0
88	MN_4	0	57.4238	100	0
89	neu18	0	57.4967	100	0
90	neu19	0	57.9271	100	0
91	M_4	0	57.9682	100	0
92	2MKS_4	0	58.0503	100	0
93	SN_4	0	58.4397	100	0
94	3MN_4	0	58.5126	100	0
95	neu20	0	58.943	100	0
96	MS_4	0	58.9841	100	0
97	MK_4	0	59.0662	100	0
98	2MSN_4	0	59.5285	100	0
99	S_4	0	60.	100	0

100	SK_4	0	60.0821	100	0
101	3MO_5	0	73.0093	100	0
102	neu22	0	84.9205	100	0
103	neu23	0	85.392	100	0
104	neu24	0	85.4649	100	0
105	4MK_6	0	85.8543	100	0
106	2NM_6	0	85.8636	100	0
107	4MS_6	0	85.9364	100	0
108	2MN_6	0	86.4079	100	0
109	neu25	0	86.4808	100	0
110	M_6	0	86.9523	100	0
111	MSN_6	0	87.4238	100	0
112	4MN_6	0	87.4967	100	0
113	MNK_6	0	87.506	100	0
114	2MS_6	0	87.9682	100	0
115	2MK_6	0	88.0503	100	0
116	2SN_6	0	88.4397	100	0
117	3MSN_6	0	88.5126	100	0
118	2SM_6	0	88.9841	100	0
119	MSK_6	0	89.0662	100	0

120	S_6	0	90.	100	0
121	neu26	0	114.376	100	0
122	5MK_8	0	114.838	100	0
123	5MS_8	0	114.921	100	0
124	3MN_8	0	115.392	100	0
125	neu27	0	115.465	100	0
126	M_8	0	115.936	100	0
127	2MSN_8	0	116.408	100	0
128	5MN_8	0	116.481	100	0
129	3MS_8	0	116.952	100	0
130	3MK_8	0	117.034	100	0
131	neu28	0	117.497	100	0
132	2{MS}_8	0	117.968	100	0
133	2MSK_8	0	118.05	100	0
134	4MN_10	0	144.376	100	0
135	M_10	0	144.921	100	0
136	neu29	0	145.392	100	0
137	4MS_10	0	145.936	100	0
138	4MK_10	0	146.019	100	0
139	neu30	0	146.481	100	0

140	neu31	0	146.952	100	0
141	M_12	0	173.905	100	0
142	5MS_12	0	174.921	100	0
143	5MN_12	0	174.966	100	0
144	5MK_12	0	175.003	100	0
145	M_14	0	202.889	100	0
146	6MS_14	0	203.905	100	0
147	M_18	0	260.857	100	0
148	M_22	0	318.825	100	0

Partialtiden - PTSynthese

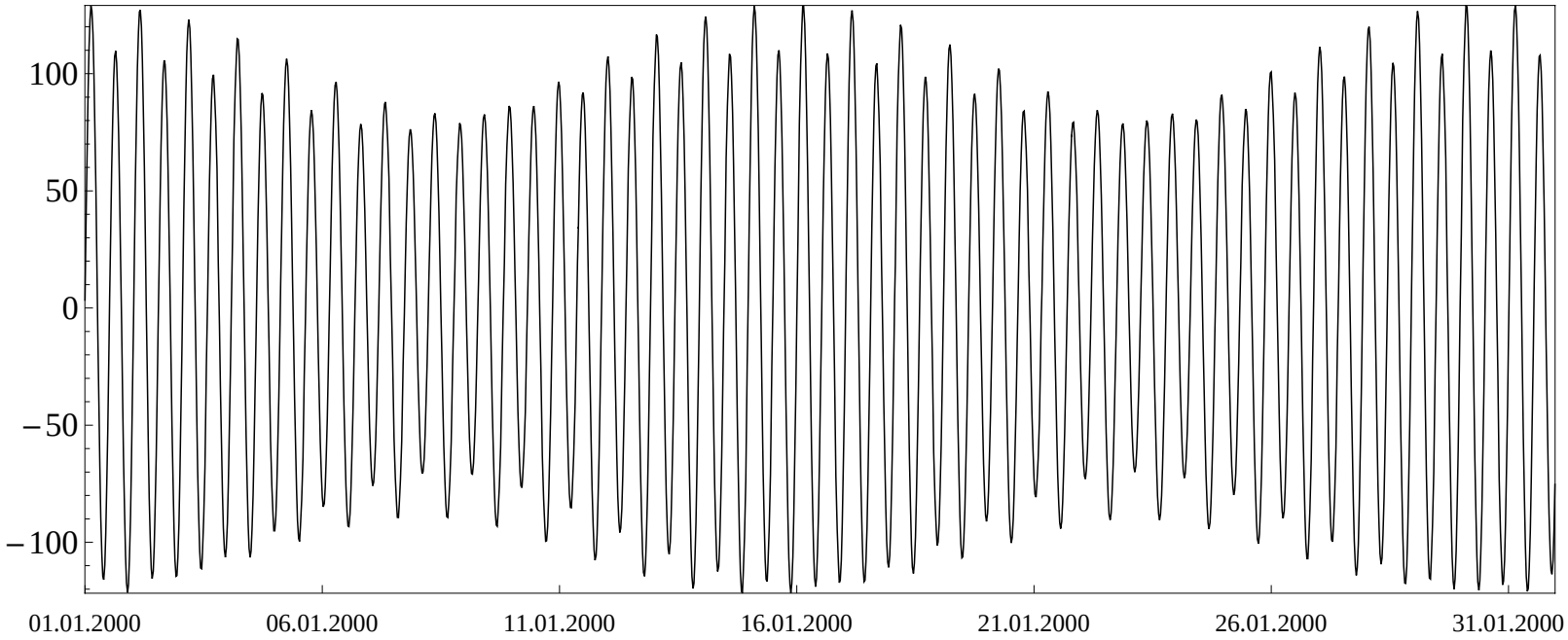
In[131]:=
```
?PTSynthese
synTS = PTSynthese[{{"M_2", 100, 0}, {"S_2", 20, 0}, {"K_1", 10, 20}},
    AbsoluteTime[{2000, 1, 1, 0, 0, 0}], Anfangszeit → AbsoluteTime[{2000, 1, 1, 0, 0, 0}],
    Endzeit → AbsoluteTime[{2000, 1, 31, 23, 59, 59}],
    Sampledauer → 30 * 60, VerboseMode → False];
TSFrame[synTS]
TSListPlotArray[synTS, ISize → 800][[1]]
```

PTSynthese[Partialtidenliste, Referenzzeitpunkt, Optionen] –> Zeitreihe(TS)
Partialtidenliste = {{PTName 1, Amplitude 1, Phase 1}, ...}
Die Funktion erzeugt eine Zeitreihe aus den Partialtiden der Partialtidenliste. Die Phase der Partialtide bezieht sich
auf den angegebenen Referenzzeitpunkt. Die Zeitspanne der Zeitreihe kann über die Optionen Anfangs– und Endzeit frei gewählt werden.

Optionen:
Anfangszeit –> AbsoluteTime[{2000, 1, 1, 0, 0, 0}]
Endzeit –> AbsoluteTime[{2000, 12, 31, 23, 30, 0}]
Sampledauer –> 30*60 (Zeitabstand zweier Datenpunkte)
VerboseMode –> False (True –> mehr Infos)

Out[133]= 01.01.2000 00:00:00 – 31.01.2000 23:30:00 ––> 1488 Datensätze. Länge: 0a30d23h30m0s

Out[134]=


100
50
0
−50
−100
01.01.2000
06.01.2000
11.01.2000
16.01.2000
21.01.2000
26.01.2000
31.01.2000

Partialtiden - Fensterfunktionen

In[135]:=
```
?WFRechteck
?WFFejer
?WFHanning
?WFHamming
?WFFlatTop
?WFGauss
?WFKaiser
?WFBlackman
?WFBlackman2
?WFTriplett
n = 100;
alphagesamt[i_, N_, beta_] := beta * Sqrt[1 - (2 * (i - (N - 1) / 2) / (N - 1))^2];
alpha[i_] = alphagesamt[i, n, 4];
GraphicsArray[
 {{ListPlot[Table[WFRechteck[], {i, 1, n}], Joined → True, PlotLabel → "Rechteck"],
   ListPlot[Table[WFFejer[i, n], {i, 1, n}], Joined → True, PlotLabel → "Fejer"],
   ListPlot[Table[WFHanning[i, n], {i, 1, n}], Joined → True, PlotLabel → "Hanning"]},
```

```
{ListPlot[Table[WFHamming[i, n, 0.1], {i, 1, n}], Joined → True, PlotLabel → "Hamming"],
 ListPlot[Table[WFFlatTop[i, n], {i, 1, n}], Joined → True, PlotLabel → "FlatTop"],
 ListPlot[Table[WFGauss[i, n, 2, 5], {i, 1, n}], Joined → True, PlotLabel → "Gauss"]},
{ListPlot[Table[WFKaiser[i, alpha[i], 4], {i, 1, n}], Joined → True, PlotLabel → "Kaiser"],
 ListPlot[Table[WFBlackman[i, n], {i, 1, n}], Joined → True, PlotLabel → "Blackman"],
 ListPlot[Table[WFBlackman2[i, n], {i, 1, n}], Joined → True, PlotLabel → "Blackman2"]},
{ListPlot[Table[WFTriplett[i, n, 4], {i, 1, n}], Joined → True, PlotLabel → "Triplett"]}},
ImageSize → 800]
```

Rechteck–Fensterfunktion []

Fejer–Fensterfunktion [i, N]

Hanning–Fensterfunktion [i, N]

Hamming–Fensterfunktion [i, N, alpha]

FlatTop–Fensterfunktion [i, N]

Gauss–Fensterfunktion [i, N, Sigma, SigmaCut]

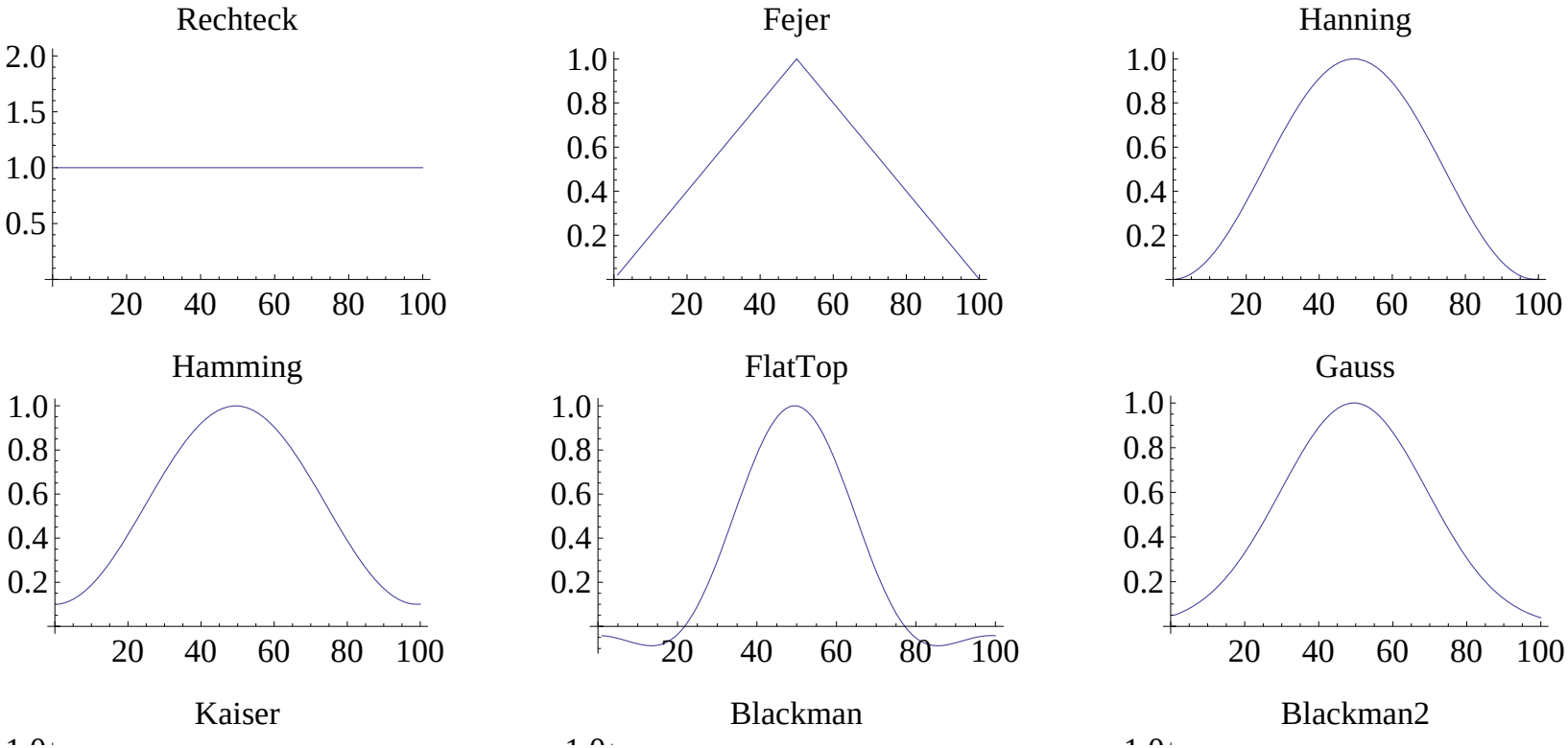
Kaiser–Fensterfunktion [i, Alpha[Beta], Beta]

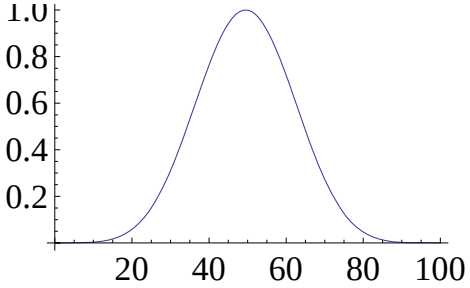
Blackman–Fensterfunktion [i, N]

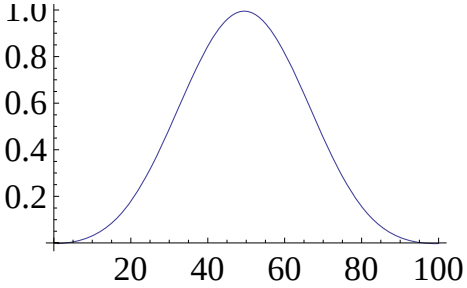
Blackman2−Fensterfunktion [i, N]

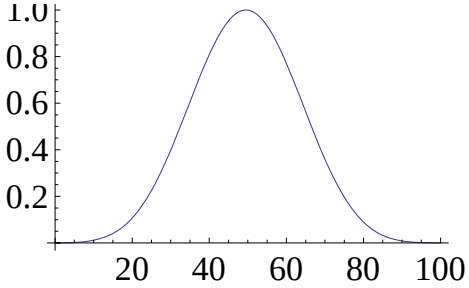
Triplett−Fensterfunktion [i, N, Gamma]

Out[148]=

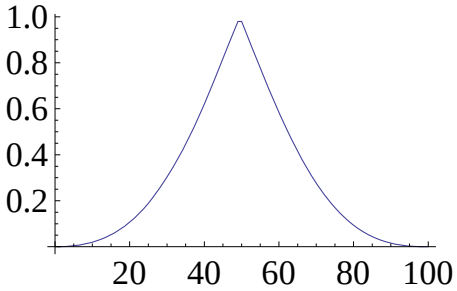



1.0
0.8
0.6
0.4
0.2
20
40
60
80
100


1.0
0.8
0.6
0.4
0.2
20
40
60
80
100


1.0
0.8
0.6
0.4
0.2
20
40
60
80
100

Triplett


1.0
0.8
0.6
0.4
0.2
20
40
60
80
100

Partialtiden - TScalcSpektrum

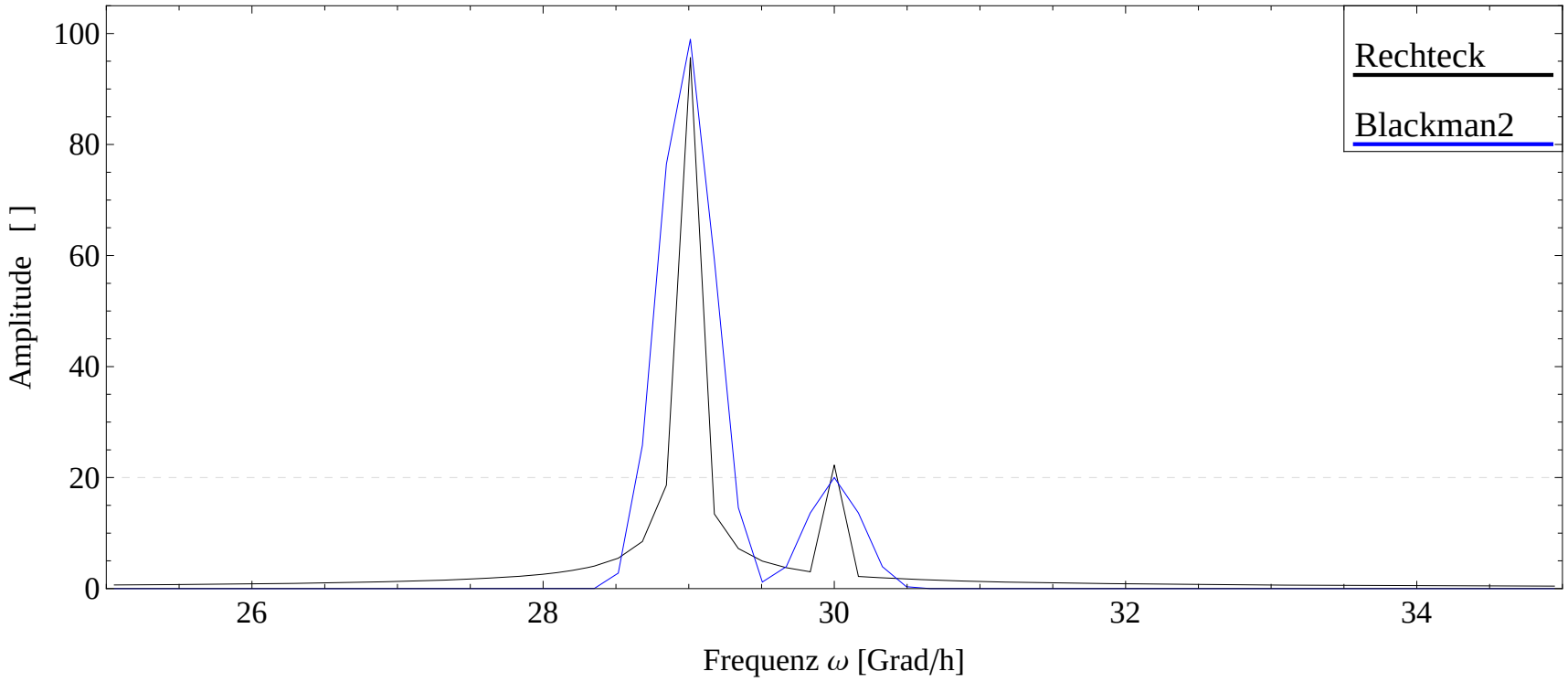
In[149]:=
```
?TScalcSpektrum
synTS = PTSynthese[{{"M_2", 100, 0}, {"S_2", 20, 0}},
   AbsoluteTime[{2000, 1, 1, 0, 0, 0}], Anfangszeit → AbsoluteTime[{2000, 1, 1, 0, 0, 0}],
   Endzeit → AbsoluteTime[{2000, 3, 31, 23, 59, 59}]];
TSFrame[synTS]
spec1 = TScalcSpektrum[synTS, WindowFunction → Rechteck,
   ZeroPaddingFactor → 1, CalcPhases → False, VerboseMode → False];
spec2 = TScalcSpektrum[synTS, WindowFunction → Blackman2,
   ZeroPaddingFactor → 1, CalcPhases → False, VerboseMode → False];
ListPlotSpektrum[{spec1, spec2}, XRange -> {25, 35}, YRange -> {0, 105},
 Threshold → 20, ColorList → {Black, Blue}, ISize → 800,
 LegendEntryList → {"Rechteck", "Blackman2"}, Legend → True]
```

TScalcSpektrum[TS, Optionen] –> {{Frequenz, Amplitude[, Phase]}, ...}
Die Funktion berechnet das Spektrum einer Zeitreihe TS.

Optionen:
WindowFunction –> Hanning (Fensterfunktionen: Rechteck, Fejer, Hanning, Hamming, FlatTop, Gauss, Kaiser, Blackman, Blackman2, Triplett)
ZeroPaddingFactor –> 1 (das n–fache an Nullen wird an die (gewichtete) Zeitreihe angehängt)
CalcPhases –> False (True –> das Phasenspektrum wird berechnet)
VerbodeMode –> False (True –> mehr Infos)

Out[151]= `01.01.2000 00:00:00 – 31.03.2000 23:30:00 –-> 4368 Datensätze. Länge: 0a90d23h30m0s`

Out[154]=

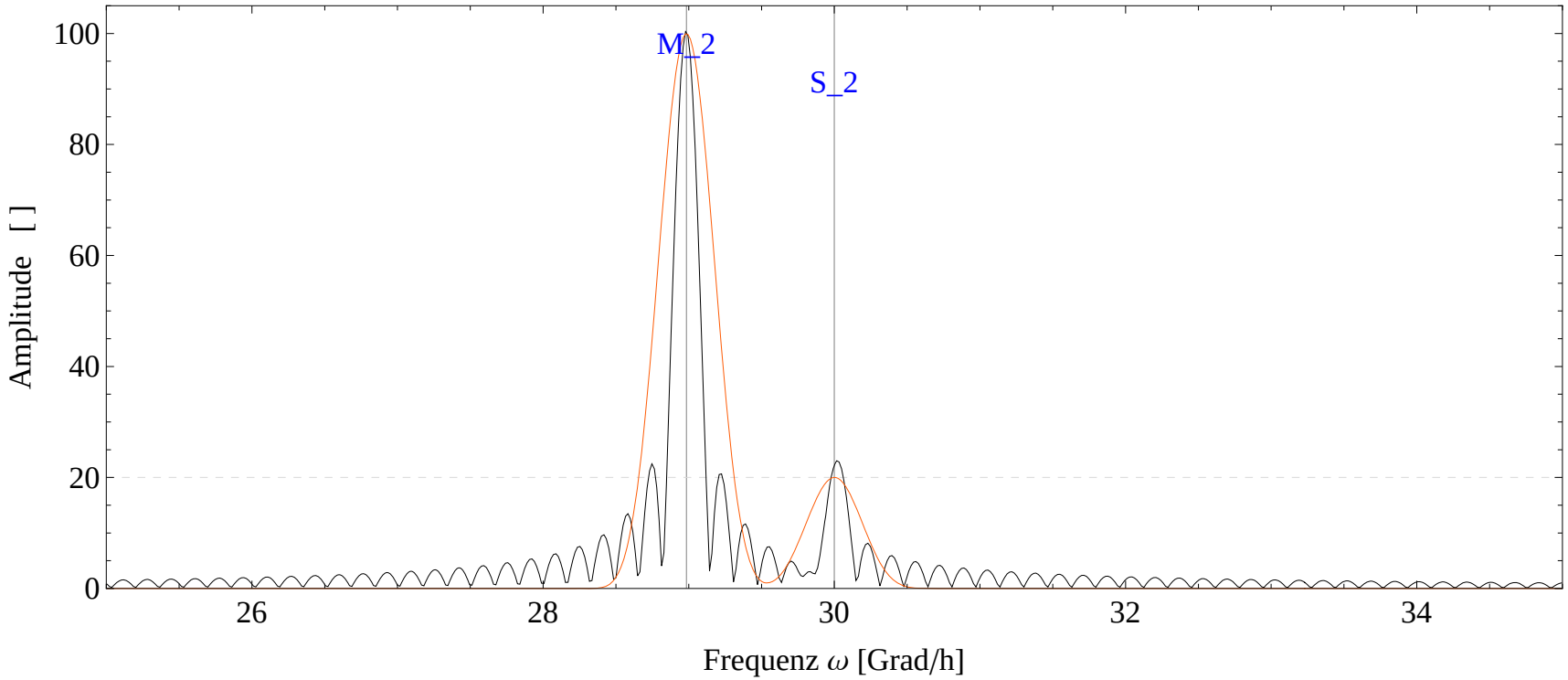

Rechteck
Blackman2
Amplitude [ ]
100
80
60
40
20
0
26
28
30
32
34
Frequenz ω [Grad/h]

◄ | ►

Partialtiden - ListPlotSpektrum

In[155]:=
```
spec3 = TScalcSpektrum[synTS, WindowFunction → Rechteck,
   ZeroPaddingFactor → 10, CalcPhases → False, VerboseMode → False];
spec4 = TScalcSpektrum[synTS, WindowFunction → Blackman2,
   ZeroPaddingFactor → 10, CalcPhases → False, VerboseMode → False];
ListPlotSpektrum[{spec3, spec4}, XRange -> {25, 35}, YRange -> {0, 105},
 Threshold → 20, showPTList → {"M_2", "S_2"}, ISize → 800]
?ListPlotSpektrum
```

Out[157]=



ListPlotSpektrum[Spektrum oder {Spektrum1, Spektrum2, ...}, Optionen] –> Grafik
Die Funktion stellt ein Spektrum graphisch dar. Das Spektrum muss als Liste in folgender Form übergeben werde: {{Frequenz 1, Wert1}, {Frequenz 2, Wert2}, ...}

Optionen:
XRange –> Automatic ({Xvon, Xbis})
YRange –> Automatic ({Yvon,Ybis}})
ColorList –> Automatic (Liste mit Farben[pro Spektrum])
ThicknessList –> Automatic (Dicke der Linien)
TextSize –> 16 (Textgröße der Achsbeschriftung)
PLabel –> (Beschriftung der Grafik)
PLabelTextSize –> 20 (Textgröße Beschriftung)
PLabelTextCol –> Black (Textfarbe Beschriftung)
FrameLabel –> {Frequenz ω [Grad/h], Amplitude [ ]} (Achsbeschriftung {x,y,x2,y2})
Legend –> False (True –> ein)
LegendEntryList –> Automatic (Legendeneinträge)
LegendLabel –> (Überschrift der Legende)
showPTList –> {} (Anzeige der Partialtiden z.B. {'M_2', 'S_2'})
showLinesList –> {} (zusätzliche Linien im Spektrum {{Name, Frequenz}, ...})
markPTList –> {} (markiert Partialtiden zusätzlich; z.B. {'M_2', 'S_2'})
MarkColor –> Blue (Farbe der Markierung)
MarkTextSize –> 16 (Textgröße der Linienbeschriftung)
ShowThreshold –> True (False –> keine Anzeige)
Threshold –> 1 (horizontale Linie)
AspecRatio –> 2/5 (Seitenverhältnis)
ISize –> 1200 (Größe der Abbildung in Pixeln)
LogPlot –> False (True –> LogPlot)

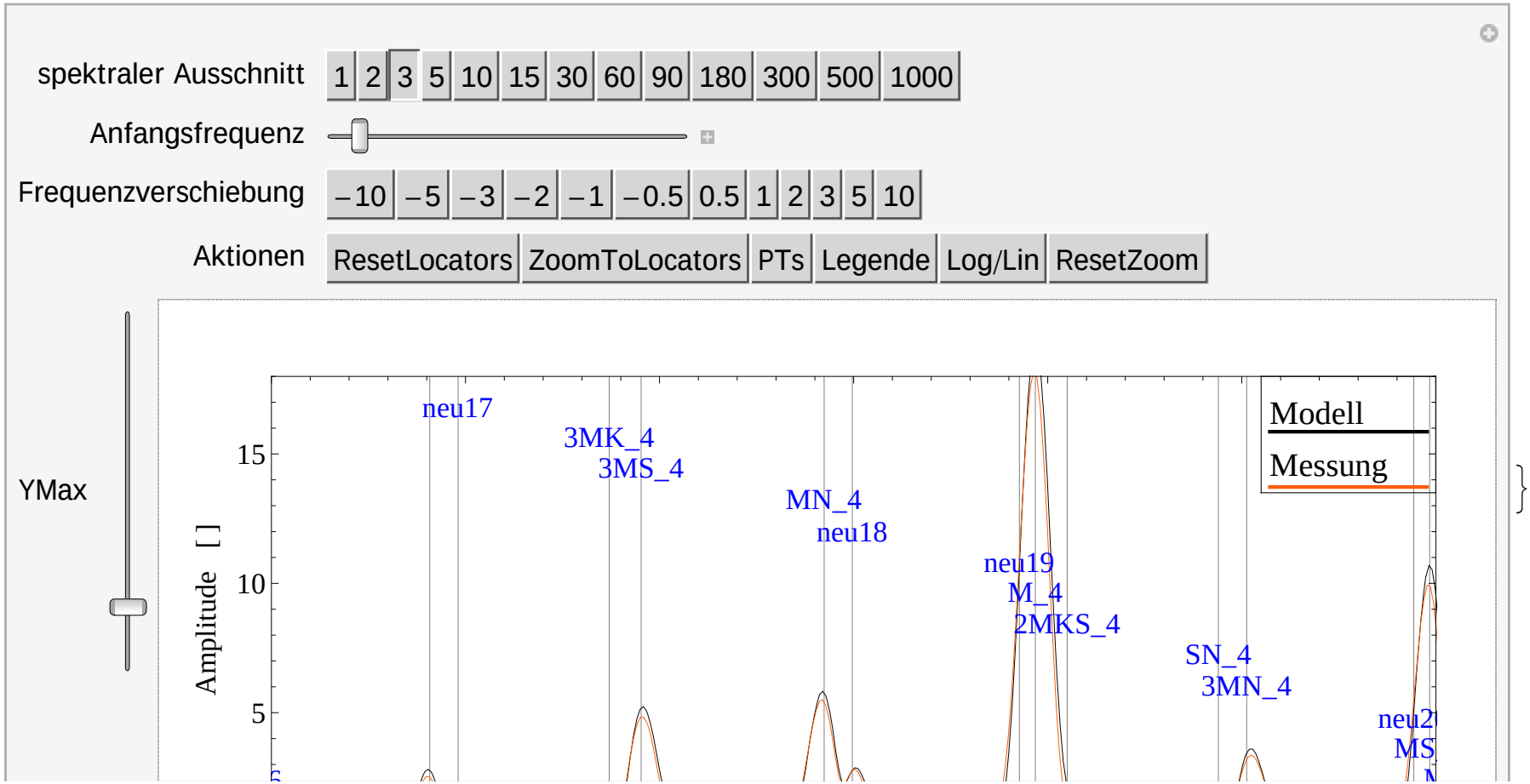
Partialtiden - ListPlotSpektrumI

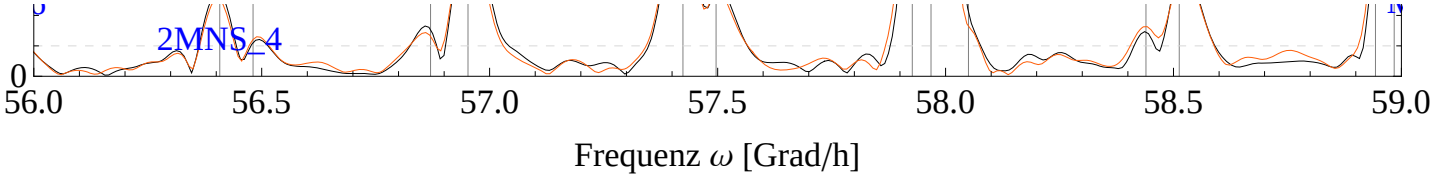
In[159]:=
```
{modellTS, nfp, name, rw, hw, anzp, phylist} = ReadBoe["D:\\Eigene
     Dateien\\Output\\Vortrag\\2009_09_25_TV12_MM7-Lib\\wl.StPauli.modell.2006.dat"];
submodellTS = TSSelect[modellTS, AbsoluteTime[{2006, 1, 2, 0, 0, 0}],
   AbsoluteTime[{2006, 12, 31, 23, 59, 59}]];
TSFrame[submodellTS]
mesTS = Get["D:\\Eigene
     Dateien\\Output\\Vortrag\\2009_09_25_TV12_MM7-Lib\\StPauli_01012006-31122006.mm"];
submesTS = TSSelect[mesTS, AbsoluteTime[{2006, 1, 2, 0, 0, 0}],
   AbsoluteTime[{2006, 12, 31, 23, 59, 59}]];
TSFrame[submesTS]
modspec = TScalcSpektrum[Transpose[{submodellTS[[All, 1]], submodellTS[[All, 2]] * 100}],
   WindowFunction → Hanning, ZeroPaddingFactor → 5];
messpec = TScalcSpektrum[submesTS, WindowFunction → Hanning, ZeroPaddingFactor → 5];
ListPlotSpektrumI[{modspec, messpec}, showPTList → ptlist[[All, 1]],
 YRange → {0, 120}, LegendEntryList → {"Modell", "Messung"}, Legend → True, ISize → 700]
```

Out[161]= 02.01.2006 00:00:00 – 31.12.2006 00:00:00 --> 52273 Datensätze. Länge: 0a363d0h0m0s

Out[164]= 02.01.2006 00:00:00 – 31.12.2006 23:50:00 --> 52416 Datensätze. Länge: 0a363d23h50m0s

Out[167]=


spektraler Ausschnitt
1 2 3 5 10 15 30 60 90 180 300 500 1000
Anfangsfrequenz
Frequenzverschiebung
−10 −5 −3 −2 −1 −0.5 0.5 1 2 3 5 10
Aktionen
ResetLocators ZoomToLocators PTs Legende Log/Lin ResetZoom
YMax
Amplitude [ ]
15
10
5
neu17
3MK_4
3MS_4
MN_4
neu18
neu19
M_4
2MKS_4
SN_4
3MN_4
Modell
Messung


2MNS_4
0
56.0
56.5
57.0
57.5
58.0
58.5
59.0
Frequenz ω [Grad/h]

}

Partialtiden - FitPT

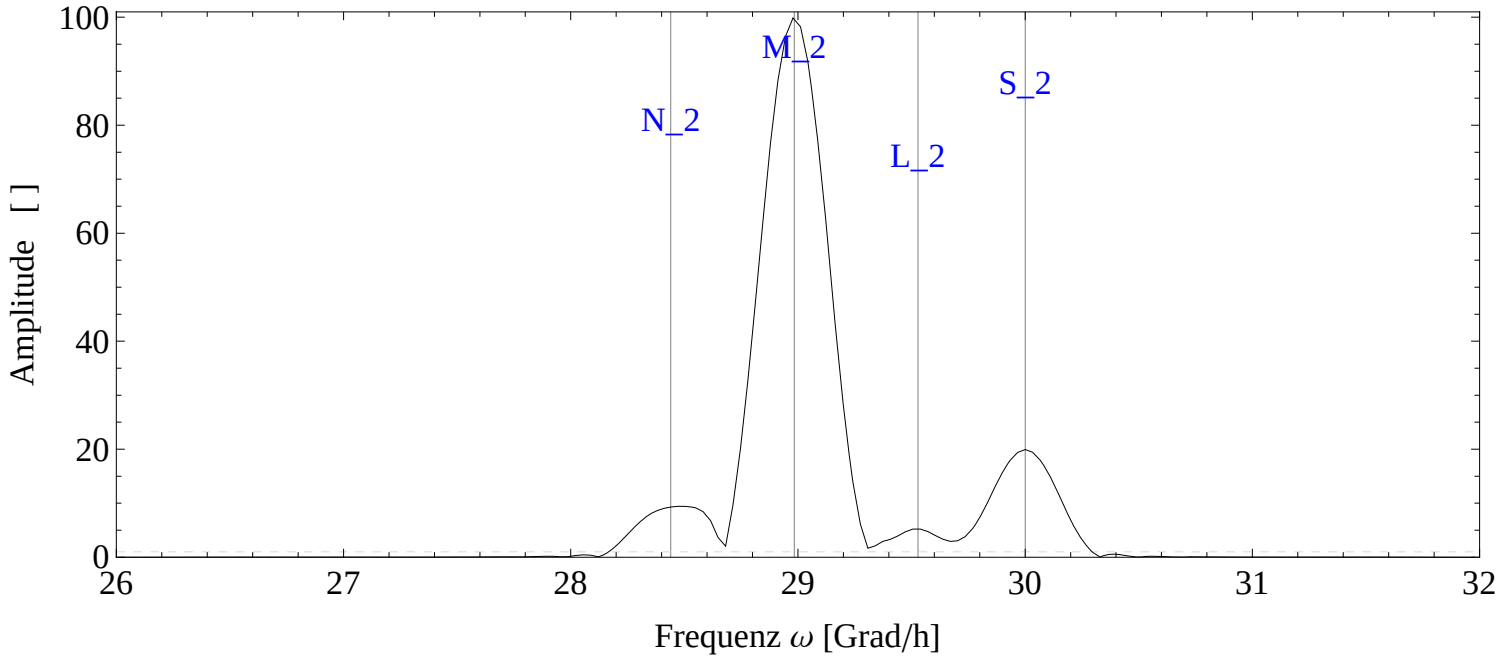
In[168]:=
```
?FitPT
synTS = PTSynthese[{{"M_2", 100, 0}, {"S_2", 20, 25}, {"N_2", 10, 30}, {"L_2", 5, 190}},
   AbsoluteTime[{2000, 1, 1, 0, 0, 0}], Anfangszeit → AbsoluteTime[{2000, 1, 1, 0, 0, 0}],
   Endzeit → AbsoluteTime[{2000, 3, 31, 23, 59, 59}], Sampledauer → 10 * 60];
TSFrame[synTS]
spec = TScalcSpektrum[synTS, WindowFunction → Hanning, ZeroPaddingFactor → 5];
ListPlotSpektrum[spec, showPTList → {"M_2", "S_2", "N_2", "L_2"},
 XRange -> {26, 32}, YRange → {0, 101}, ISize → 700]
resultfit = FitPT[synTS, {"M_2", "S_2", "L_2"}];
TableForm[resultfit,
 TableHeadings → {{}, {"Name", "Frequenz [Grad/h]", "Amplitude", "Phase [Grad]"}}]
```

FitPT [TS, SuchtidenListe, Optionen] –> {{Name, Frequenz[Grad/h], Amplitude, Phase[2*Pi!!!]}, ...}
Die Funktion bestimmt die Amplituden und Phasen der gesuchten Partialtiden über ein Regressionsverfahren.
SuchtidenListe –> {M_2, S_2, ...}

Optionen:
MaxIterations –> 1000 (ohne Funktion)
VerboseMode –> False (True –> mehr Infos)

Out[170]= `01.01.2000 00:00:00 - 31.03.2000 23:50:00 -–> 13104 Datensätze. Länge: 0a90d23h50m0s`

Out[172]=



Out[174]//TableForm=

Name	Frequenz [Grad/h]	Amplitude	Phase [Grad]
L_2	29.5285	−5.00103	1.02526
M_2	28.9841	−100.678	13.4333
S_2	30.	−20.0758	−2.72138

Partialtiden - FFTPT

In[175]:=
```
?FFTPT
resultfft = FFTPT[synTS, {"M_2", "S_2", "L_2"}, ZeroPaddingFactor → 5];
TableForm[resultfft,
 TableHeadings → {{}, {"Name", "Frequenz [Grad/h]", "Amplitude", "Phase [Grad]"}}]
```

FFTPT[TS, SuchtidenListe, Optionen] –> {{Name, Frequenz[Grad/h], Amplitude, Phase[Grad], ErrorFrequenz[Grad/h], –, –, ErrorFrequenzTolerance}, ...}
Die Funktion bestimmt die Amplituden und Phasen der gesuchten Partialtiden über eine FFT.
SuchtidenListe –> {M_2, S_2, ...}

Optionen:
WindowFunction –> Hanning (Fensterfunktion; Auswahl siehe TScalcSpektrum)
ZeroPaddingFactor –> 10 (Die Anzahl der Nullen, die angehängt werden. Dieser
    Faktor (N) bestimmt die N–fache Länge der Zeitreihe. Ein Faktor von 2 verdoppelt damit die Länge der Zeitreihe.)
ErrorFrequencyTolerance –> 0.0002 (Übersteigt der Wert von ErrorFrequenz den Wert von ErrorFrequencyTolerance, wird das Flag gesetzt und rot markiert. ErrorFrequenz
    ist die Differenz zwischen der gesuchten PT–Frequenz und der der gesuchten Frequenz am nächsten kommenden Frequenz aus dem diskreten Spektrum.)
VerboseMode –> False (True –> mehr Infos)

Out[177]//TableForm=

Name	Frequenz [Grad/h]	Amplitude	Phase [Grad]				
M_2	28.9841	99.8526	6.67281	0.00608226	0	0	1
S_2	30.	19.927	25.1534	0.	0	0	0
L_2	29.5285	5.20723	171.068	–0.00998259	0	0	1

Partialtiden - FFTPTScan

In[178]:=
```
?FFTPTScan
resultfftscan = FFTPTScan[synTS, {"M_2", "S_2", "L_2"}, WindowFunction → Hanning];
TableForm[resultfftscan,
 TableHeadings → {{}, {"Name", "Frequenz [Grad/h]", "Amplitude", "Phase [Grad]"}}]
```

FFTPTScan[TS, SuchtidenListe, Optionen] –> {{Name, Frequenz[Grad/h], Amplitude, Phase[Grad],
Frequenzfehler[Grad/h], Wellenzahl (nur gültig bei no–zero–padding), eingefügte Nullen, Rückgabewert (komplex)}, ...}
Die Funktion bestimmt die Amplituden und Phasen der gesuchten Partialtiden über eine FFT. Zuvor wird die optimale einzufügende Anzahl von Nullen (zero padding) berechnet und zwar so, dass die optimale Abtastfrequenz (Partialtidenfrequenz) erreicht wird. In Frequenzfehler wird die Abweichung der Partialtidenfrequenz von der tatsächlich abgetasteten Frequenz zurückgegeben. Das Abbruchkriterium ist der Parameter MaxZeroes, der das Maximum an einzufügenden Nullen vorgibt.
SuchtidenListe –> {M_2, S_2, ...}

Optionen:
WindowFunction –> Hanning (Fensterfunktion [Rechteck, Fejer, Hanning, Hamming, FlatTop, Gauss, Kaiser, Blackman, Blackman2, Triplett])
MaxZeroes –> 100000 (Vorgabe der maximalen Anzahl einzufügender Nullen; Abbruchkriterium)
VerboseMode –> False (True –> mehr Infos)

Out[180]//TableForm=

Name	Frequenz [Grad/h]	Amplitude	Phase [Grad]				
M_2	28.9841	99.9371	0.0319756	-8.04561×10^{-7}	233	4260	$0.0557728+99.9371$
S_2	30.	19.927	25.1534	0.	185	216	$8.46986+18.0374\,i$
L_2	29.5285	5.25938	182.337	-1.80846×10^{-7}	895	52 365	$-0.214474-5.25501$

Partialtiden - sFFTPTScan

In[181]:=
```
?sFFTPTScan
resultsfftscan = sFFTPTScan[synTS, {"M_2", "S_2", "L_2"}, WindowFunction → Hanning];
TableForm[resultsfftscan,
 TableHeadings → {{}, {"Name", "Frequenz [Grad/h]", "Amplitude", "Phase [Grad]"}}]
```

sFFTPTScan[TS, SuchtidenListe, Optionen] –> {{Name, Frequenz[Grad/h], Amplitude, Phase[Grad],
ErrorFrequenz[Grad/h], Wellenzahl (nur gültig bei no–zero–padding), eingefügte Nullen, Rückgabewert (komplex)}, ...}
Die Funktion bestimmt die Amplituden und Phasen der gesuchten Partialtiden sukzessiv über eine FFT (s. FFTPTScan). Nach jeder Analyse wird die Partialtide mit der größten Amplitude bestimmt. Diese wird aus dem Signal entfernt und die Analyse geht weiter. Damit werden kleine Partialtiden, die auf Schultern von sehr nahen und sehr hohen Partialtiden sitzen, realistischer detektiert als ohne die Elimination.
SuchtidenListe –> {M_2, S_2, ...}

Optionen:
WindowFunction –> Hanning (Fensterfunktion [Rechteck, Fejer, Hanning, Hamming, FlatTop, Gauss, Kaiser, Blackman, Blackman2, Triplett])
MaxZeroes –> 100000 (Vorgabe der maximalen Anzahl einzufügender Nullen; Abbruchkriterium)
VerboseMode –> False (True –> mehr Infos)

Out[183]//TableForm=

Name	Frequenz [Grad/h]	Amplitude	Phase [Grad]				
M_2	28.9841	99.9371	0.0319756	-8.04561×10^{-7}	233	4260	$0.0557728+99.9371\,i$
S_2	30.	19.9669	24.9942	0.	185	216	$8.43652+18.097\,i$
L_2	29.5285	4.99983	189.87	-1.80846×10^{-7}	895	52 365	$-0.857049-4.92583\,i$

Partialtiden - sFFTPTScanCluster

In[184]:=
```
?sFFTPTScanCluster
resultsfftscancluster =
  sFFTPTScanCluster[synTS, {{"M_2", "S_2", "L_2"}}, WindowFunction → Hanning];
TableForm[resultsfftscancluster, TableHeadings →
  {{}, {"Name", "Frequenz [Grad/h]", "Amplitude", "Phase [Grad]"}}]
```

sFFTPTScanCluster [TS, SuchtidenListe, Optionen] –> {{Name, Frequenz[Grad/h], Amplitude,
Phase[Grad], ErrorFrequenz[Grad/h], Wellenzahl (nur gültig bei no–zero–padding), eingefügte Nullen, Rückgabewert (komplex)}, ...}
Die Funktion bestimmt die Amplituden und Phasen der gesuchten Partialtiden sukzessiv über eine FFT (s. sFFTPTScan). Die Analyse folgt Cluster für Cluster.
SuchtidenListe –> {Cluster1, Cluster2, ..., Cluster N}; Cluster1..N = {PT1, PT2, ... PT M}

Optionen:
WindowFunction –> Hanning (Fensterfunktion [Rechteck, Fejer, Hanning, Hamming, FlatTop, Gauss, Kaiser, Blackman, Blackman2, Triplett])
MaxZeroes –> 100000 (Vorgabe der maximalen Anzahl einzufügender Nullen; Abbruchkriterium)
VerboseMode –> False (True –> mehr Infos)

Out[186]//TableForm=

Name	Frequenz [Grad/h]	Amplitude	Phase [Grad]				
M_2	28.9841	99.9371	0.0319756	-8.04561×10^{-7}	233	4260	$0.0557728+99.9371$
S_2	30.	19.9669	24.9942	0.	185	216	$8.43652+18.097\,i$
L_2	29.5285	4.99983	189.87	-1.80846×10^{-7}	895	52 365	$-0.857049-4.92583$

Partialtiden - sFFTPTScanCluster

In[187]:=
```
res = sFFTPTScanCluster[mesTS,
      {{"Q_1"}, {"O_1"}, {"M_1"}, {"P_1", "S_1", "K_1"}, {"neu04"}, {"neu05"}, {"neu07"},
       {"3M{SK}_2", "3M2S_2"}, {"OQ_2", "EPS_2", "neu09"}, {"O_2", "MU_2", "neu11"},
       {"N_2", "NU_2"}, {"unnamed1", "unnamed2", "M_2", "unnamed3", "DELTA_2", "neu12"},
       {"LABDA_2", "neu13", "L_2"}, {"T_2", "S_2", "R_2", "K_2"}, {"MSN_2"},
       {"2SM_2", "neu14"}, {"NO_3"}, {"MO_3"}, {"M_3"}, {"MP_3", "MK_3"}, {"neu15", "neu16"},
       {"2MNS_4", "neu17"}, {"3MK_4", "3MS_4"}, {"MN_4", "neu18"}, {"neu19", "M_4"},
       {"SN_4", "3MN_4"}, {"neu20", "MS_4", "MK_4"}, {"2MSN_4"}, {"S_4", "SK_4"},
       {"3MO_5"}, {"neu22"}, {"neu23", "neu24"}, {"4MK_6", "4MS_6"}, {"2MN_6", "neu25"},
       {"M_6"}, {"MSN_6", "4MN_6"}, {"2MS_6", "2MK_6"}, {"2SN_6", "3MSN_6"},
       {"2SM_6", "MSK_6"}, {"neu26"}, {"5MK_8", "5MS_8"}, {"3MN_8", "neu27"}, {"M_8"},
       {"2MSN_8", "5MN_8"}, {"3MS_8", "3MK_8"}, {"neu28"}, {"2{MS}_8", "2MSK_8"},
       {"4MN_10"}, {"M_10"}, {"neu29"}, {"4MS_10", "4MK_10"}, {"neu30"}, {"neu31"},
       {"M_12"}, {"5MS_12"}}, WindowFunction → Hanning]; // AbsoluteTiming
TableForm[res, TableHeadings → {{}, {"Name", "Frequenz [Grad/h]",
    "Amplitude", "Phase [Grad]"}}]
```

Out[187]= {111.3238500, Null}

Out[188]//TableForm=

Name	Frequenz [Grad/h]	Amplitude	Phase [Grad]				
Q_1	13.3987	3.39061	187.74	-3.4388×10^{-7}	809	77 859	−0.456644 − 3.
O_1	13.943	10.8732	89.2164	-5.40662×10^{-8}	405	10 181	10.8722 + 0.14
M_1	14.4921	0.685433	356.875	-4.0228×10^{-7}	466	16 896	−0.0373606 + 0
K_1	15.0411	8.10904	349.788	-2.46302×10^{-7}	763	57 012	−1.43768 + 7.9
P_1	14.9589	3.48024	315.509	-1.57044×10^{-6}	1027	95 734	−2.43892 + 2.4
S_1	15.	1.07549	171.778	−0.0000964024	1059	99 935	0.153806 − 1.0
neu04	25.9364	1.54204	286.408	-2.38809×10^{-7}	752	10 067	−1.47925 + 0.4
neu05	26.3258	1.73442	280.065	-1.88099×10^{-6}	656	1264	−1.70772 + 0.3
neu07	26.4808	1.92007	27.4911	-1.3597×10^{-7}	846	16 447	0.886325 + 1.7
3M2S_2	26.9523	5.33018	54.8804	-3.84789×10^{-8}	1307	52 185	4.35984 + 3.06
3M{SK}_2	26.8702	3.73173	215.53	-3.26761×10^{-7}	1855	96 557	−2.16864 − 3.0
EPS_2	27.4238	6.48794	260.45	-5.29641×10^{-8}	1709	82 047	−6.39803 − 1.0
neu09	27.4967	4.2892	163.139	-2.23021×10^{-8}	1566	70 457	1.24408 − 4.10
OQ_2	27.3417	2.36796	59.0464	−0.0000756573	666	54	2.03072 + 1.21
MU_2	27.9682	21.7122	191.61	-4.72294×10^{-8}	1375	53 632	−4.36955 − 21.
O_2	27.8861	6.58508	358.26	-1.08132×10^{-7}	810	10 181	−0.199933 + 6.

neu11	28.0411	1.11849	268.409	-4.12581×10^{-7}	736	4134	$-1.11806-0.0$
N_2	28.4397	23.9905	9.71732	-9.41988×10^{-8}	1122	32656	$4.0493+23.64$
NU_2	28.5126	10.962	291.229	-1.3918×10^{-7}	1701	76301	$-10.2181+3.9$
M_2	28.9841	157.147	305.969	-8.04561×10^{-7}	932	16896	$-127.184+92.$
unnamed1	28.902	6.9764	272.906	-1.01297×10^{-7}	1421	53639	$-6.96743+0.3$
unnamed2	28.943	2.61522	75.483	-1.48439×10^{-7}	1867	86773	$2.53173+0.65$
DELTA_2	29.0662	2.06505	329.119	-2.56784×10^{-7}	853	10829	$-1.05991+1.7$
unnamed3	29.0252	1.16563	269.554	-2.38352×10^{-7}	1729	76109	$-1.16559-0.0$
neu12	29.1391	0.151266	55.6401	-3.27619×10^{-8}	959	18528	$0.124871+0.0$
L_2	29.5285	16.2265	75.2171	-1.80846×10^{-7}	895	12909	$15.6894+4.14$
LABDA_2	29.4556	4.22482	176.899	-1.54774×10^{-6}	1981	92708	$0.228549-4.2$
neu13	29.4874	0.430479	83.9938	-9.58615×10^{-10}	1562	61859	$0.428116+0.0$
S_2	30.	36.9796	253.858	0.	740	720	$-35.5217-10.$
K_2	30.0821	15.4773	101.136	-1.58407×10^{-7}	1643	65413	$15.1859-2.98$
T_2	29.9589	1.03333	272.349	-7.81036×10^{-8}	1226	35833	$-1.03246+0.0$
R_2	30.0411	0.5754	39.2287	-7.93622×10^{-7}	1971	89158	$0.363892+0.4$
MSN_2	30.5444	2.71527	25.2876	-8.74774×10^{-8}	1543	56556	$1.15986+2.45$

2SM_2	31.0159	4.35473	49.2427	-2.89433×10^{-7}	2177	99050	3.29863 + 2.84
neu14	31.098	3.21424	248.553	-2.88869×10^{-7}	959	14050	−2.99167 − 1.1
NO_3	42.3828	1.45179	115.536	-8.78765×10^{-8}	2675	83769	1.30996 − 0.62
MO_3	42.9271	3.10714	42.6149	-1.24159×10^{-7}	2621	79323	2.10374 + 2.28
M_3	43.4762	0.373691	150.825	-1.20684×10^{-6}	1165	5320	0.182165 − 0.3
MK_3	44.0252	2.07002	304.697	-2.76993×10^{-8}	2944	91881	−1.70192 + 1.1
MP_3	43.943	1.33331	317.811	-1.20369×10^{-6}	1100	1510	−0.895415 + 0.
neu16	55.9364	1.38035	19.7554	-9.44587×10^{-8}	1375	536	0.466566 + 1.2
neu15	55.8543	0.785054	210.102	-2.74268×10^{-7}	1360	34	−0.393738 − 0.
2MNS_4	56.4079	2.52226	229.601	-1.06054×10^{-7}	3778	92109	−1.92084 − 1.6
neu17	56.4808	1.31937	129.221	-1.51221×10^{-8}	1411	1401	1.02213 − 0.83
3MS_4	56.9523	4.84957	156.81	-3.26744×10^{-8}	3903	95467	1.90965 − 4.45
3MK_4	56.8702	1.33167	353.434	-3.77117×10^{-7}	3892	95263	−0.152277 + 1.
MN_4	57.4238	5.42832	322.922	-3.81728×10^{-7}	4055	99969	−3.27271 + 4.3
neu18	57.4967	2.87876	251.76	-3.01833×10^{-8}	2471	40269	−2.73411 − 0.9
M_4	57.9682	18.1891	258.391	-1.60912×10^{-6}	2563	42942	−17.8169 − 3.6
neu19	57.9271	2.61166	30.4776	-9.03914×10^{-7}	1596	6952	1.32464 + 2.25

3MN_4	58.5126	3.19355	46.6081	-1.64218×10^{-7}	2015	21824	2.32066 + 2.19
SN_4	58.4397	1.74498	237.614	-2.84257×10^{-8}	2317	33079	−1.47357 − 0.9
MS_4	58.9841	9.9491	211.844	-1.33365×10^{-7}	1687	9218	−5.2493 − 8.45
MK_4	59.0662	4.501	55.6433	-6.94376×10^{-9}	3885	89511	3.71576 + 2.54
neu20	58.943	1.12104	319.93	-2.90232×10^{-7}	1989	20328	−0.721632 + 0.
2MSN_4	59.5285	2.26669	348.099	-1.67325×10^{-7}	2802	49111	−0.467447 + 2.
S_4	60.	0.847535	85.1113	0.	1480	720	0.844452 + 0.0
SK_4	60.0821	0.541873	317.213	-1.82416×10^{-7}	2357	32176	−0.368079 + 0.
3MO_5	73.0093	0.740234	4.8101	-2.46037×10^{-7}	4321	75278	0.0620712 + 0.
neu22	84.9205	1.01587	48.7242	-3.199×10^{-7}	3522	37024	0.763471 + 0.6
neu23	85.392	2.93193	248.132	-4.46109×10^{-7}	5466	85703	−2.72097 − 1.0
neu24	85.4649	1.3167	168.222	-7.33649×10^{-7}	5970	98323	0.268774 − 1.2
4MS_6	85.9364	3.55397	178.048	-9.44588×10^{-8}	4753	66906	0.121055 − 3.5
4MK_6	85.8543	0.396262	54.3995	-1.07381×10^{-7}	4619	63649	0.322199 + 0.2
2MN_6	86.4079	4.52614	359.303	-2.43233×10^{-7}	2177	1860	−0.0550868 + 4
neu25	86.4808	2.35442	295.174	-4.11156×10^{-8}	2569	11605	−2.13081 + 1.0
M_6	86.9523	8.84966	290.321	-2.41368×10^{-6}	2330	5320	−8.29889 + 3.0

4MN_6	87.4967	2.06262	89.4649	-1.31432×10^{-6}	2202	1800	2.06253 + 0.01
MSN_6	87.4238	1.71556	300.898	-7.53613×10^{-8}	4833	66 850	−1.4721 + 0.88
2MS_6	87.9682	8.23619	241.877	-3.74169×10^{-7}	4786	64 957	−7.2638 − 3.88
2MK_6	88.0503	3.43635	88.8708	-5.53987×10^{-8}	5888	91 881	3.43569 + 0.06
3MSN_6	88.5126	3.16899	8.18684	-7.42961×10^{-8}	4411	55 083	0.45127 + 3.13
2SN_6	88.4397	0.795496	117.28	-1.50095×10^{-7}	4126	48 211	0.707017 − 0.3
2SM_6	88.9841	1.27738	163.959	-9.16179×10^{-8}	4792	63 761	0.352965 − 1.2
MSK_6	89.0662	1.1597	28.1566	-3.32054×10^{-6}	6204	97 897	0.547242 + 1.0
neu26	114.376	1.46534	210.856	-3.3789×10^{-6}	8030	99 087	−0.751549 − 1.
5MS_8	114.921	1.37434	137.38	-2.46801×10^{-8}	7725	92 636	0.930604 − 1.0
5MK_8	114.838	0.382896	57.8805	-2.72687×10^{-8}	4844	38 551	0.32429 + 0.20
3MN_8	115.392	1.82024	310.075	-6.82571×10^{-8}	4847	38 170	−1.39285 + 1.1
neu27	115.465	0.863815	261.483	-1.40645×10^{-7}	4911	39 310	−0.854289 − 0.
M_8	115.936	2.92579	241.895	-3.21824×10^{-6}	5126	42 942	−2.5808 − 1.37
2MSN_8	116.408	1.7367	271.127	-2.33346×10^{-8}	6485	67 772	−1.73637 + 0.0
5MN_8	116.481	0.595823	95.6413	-1.02002×10^{-7}	4141	24 230	0.592937 − 0.0
3MS_8	116.952	3.86467	203.912	-1.13256×10^{-8}	5140	42 371	−1.56647 − 3.5

3MK_8	117.034	1.40479	38.5397	-4.96982×10^{-10}	7757	90604	0.875261 + 1.0
neu28	117.497	1.85478	340.1	-8.99328×10^{-8}	5296	44799	−0.631332 + 1.
2{MS}_8	117.968	1.17883	145.087	-2.66731×10^{-7}	3374	9218	0.674679 − 0.9
2MSK_8	118.05	1.17229	3.45796	-5.53987×10^{-8}	7017	75832	0.0707082 + 1.
4MN_10	144.376	0.795727	291.469	-1.33213×10^{-7}	7651	61906	−0.740517 + 0.
M_10	144.921	0.968841	233.354	-4.7281×10^{-7}	4702	17522	−0.777342 − 0.
neu29	145.392	1.23344	275.135	-8.40222×10^{-8}	5647	31334	−1.22849 + 0.1
4MS_10	145.936	1.78038	198.94	-9.44587×10^{-8}	5381	27084	−0.577872 − 1.
4MK_10	146.019	0.621498	18.9813	-1.47567×10^{-7}	6631	45530	0.202148 + 0.5
neu30	146.481	1.11839	348.261	-2.00341×10^{-7}	9160	82513	−0.227536 + 1.
neu31	146.952	0.883899	148.837	-6.28357×10^{-8}	9600	88547	0.457397 − 0.7
M_12	173.905	0.424921	211.755	-4.82736×10^{-6}	4427	2426	−0.22363 − 0.3
5MS_12	174.921	0.892314	179.121	-3.199×10^{-7}	5441	14628	0.013683 − 0.8

Beispielanwendung - Einflüsse der einzelnen Partialtiden auf die Form der Tidekurve

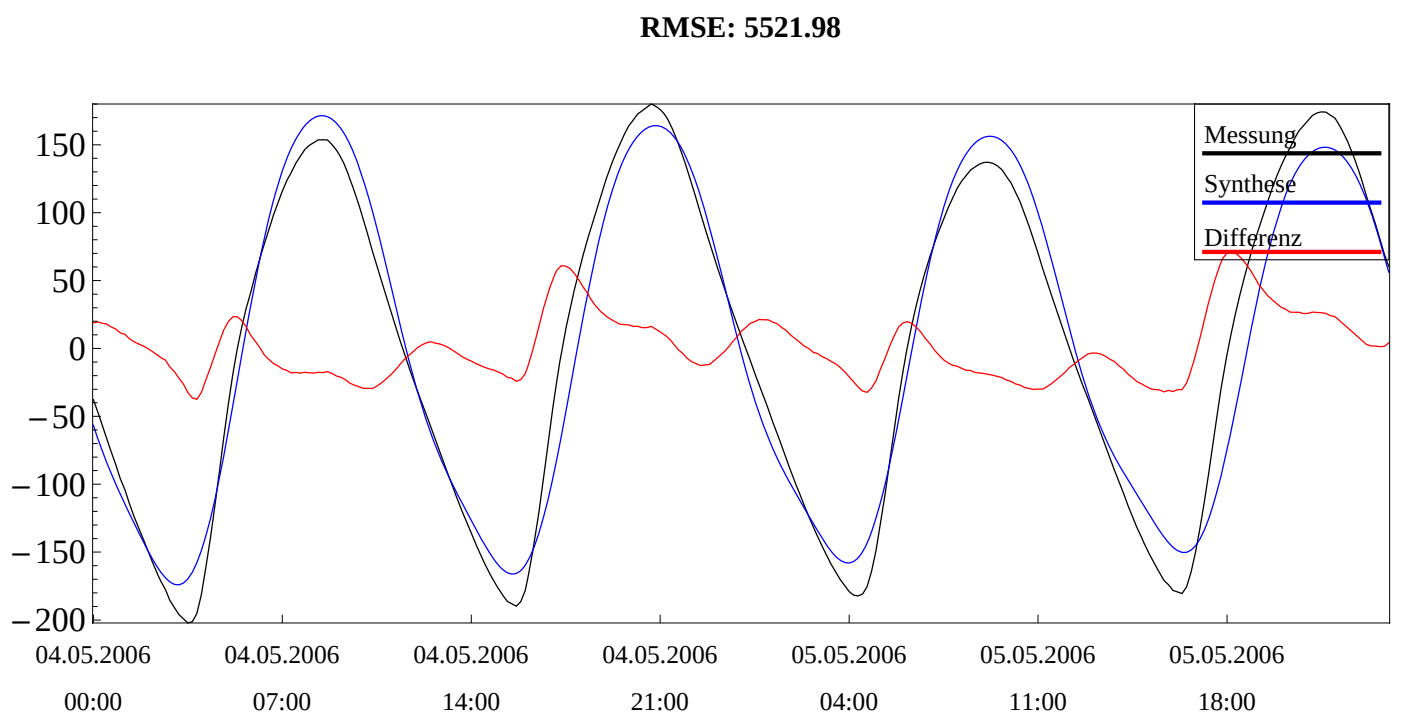
In[203]:=
```
anf = AbsoluteTime[{2006, 5, 4, 0, 0, 0}];
endz = AbsoluteTime[{2006, 5, 6, 0, 0, 0}];
pt1 = {"M_2"};(* Initialisierung: {"M_2"} oder res[[All,1]] *)
zeigept = res[[Ordering[res[[All, 3]], All, Greater]]];
showdiff = True;
(* --- *)
MW = TSLowPassFilter[mesTS, 13];
mesohneMWTS = Transpose[{mesTS[[All, 1]], mesTS[[All, 2]] - MW[[All, 2]]}];
tmpTS = TSSelect[mesohneMWTS, anf, endz];
(* --- *)
CheckboxBar[Dynamic[pt1], zeigept[[All, 1]]]
(* --- *)
Manipulate[
 ptsynthesis =
  Table[Select[zeigept[[All, {1, 3, 4}]], (#[[1]] == pt1[[i]]) &][[1]], {i, Length[pt1]}];
 synTS = PTSynthese[ptsynthesis, mesTS[[1, 1]], Anfangszeit → anf,
   Endzeit → endz, Sampledauer → 10 * 60];
 diffTS = Transpose[{tmpTS[[All, 1]], tmpTS[[All, 2]] - synTS[[All, 2]]}];
 rmse = Plus @@ Abs[diffTS[[All, 2]]];
 sarr = If[showdiff, {tmpTS, synTS, diffTS}, {tmpTS, synTS}];
 TSListPlotArray[sarr,
   LegendEntryList → Take[{"Messung", "Synthese", "Differenz"}, Length[sarr]],
   ColorList → {Black, Blue, Red}, PLabel → "RMSE: " <> ToString[rmse], ISize → 700][[1]]
 , {{showdiff, False, "Zeige Differenz"}, {True, False}}]
```

Out[211]=

☑ M_2 ☑ S_2 ☐ N_2 ☐ MU_2 ☑ M_4 ☐ L_2 ☐ K_2 ☐ NU_2 ☐ O_1 ☐ MS_4 ☑ M_6 ☐ 2MS_6
☐ K_1 ☐ unnamed1 ☐ O_2 ☐ EPS_2 ☐ MN_4 ☐ 3M2S_2 ☐ 3MS_4 ☐ 2MN_6 ☐ MK_4 ☐ 2SM_2
☐ neu09 ☐ LABDA_2 ☐ 3MS_8 ☐ 3M{SK}_2 ☐ 4MS_6 ☐ P_1 ☐ 2MK_6 ☐ Q_1 ☐ neu14
☐ 3MN_4 ☐ 3MSN_6 ☐ MO_3 ☐ neu23 ☐ M_8 ☐ neu18 ☐ MSN_2 ☐ unnamed2 ☐ neu19
☐ 2MNS_4 ☐ OQ_2 ☐ neu25 ☐ 2MSN_4 ☐ MK_3 ☐ DELTA_2 ☐ 4MN_6 ☐ neu07 ☐ neu28
☐ 3MN_8 ☐ 4MS_10 ☐ SN_4 ☐ 2MSN_8 ☐ neu05 ☐ MSN_6 ☐ neu04 ☐ neu26 ☐ NO_3 ☐ 3MK_8
☐ neu16 ☐ 5MS_8 ☐ MP_3 ☐ 3MK_4 ☐ neu17 ☐ neu24 ☐ 2SM_6 ☐ neu29 ☐ 2{MS}_8
☐ 2MSK_8 ☐ unnamed3 ☐ MSK_6 ☐ neu20 ☐ neu11 ☐ neu30 ☐ S_1 ☐ T_2 ☐ neu22
☐ M_10 ☐ 5MS_12 ☐ neu31 ☐ neu27 ☐ S_4 ☐ 4MN_10 ☐ 2SN_6 ☐ neu15 ☐ 3MO_5 ☐ M_1
☐ 4MK_10 ☐ 5MN_8 ☐ R_2 ☐ SK_4 ☐ neu13 ☐ M_12 ☐ 4MK_6 ☐ 5MK_8 ☐ M_3 ☐ neu12

Out[212]=

Zeige Differenz ☑



◄ | ►

Befehlsübersicht

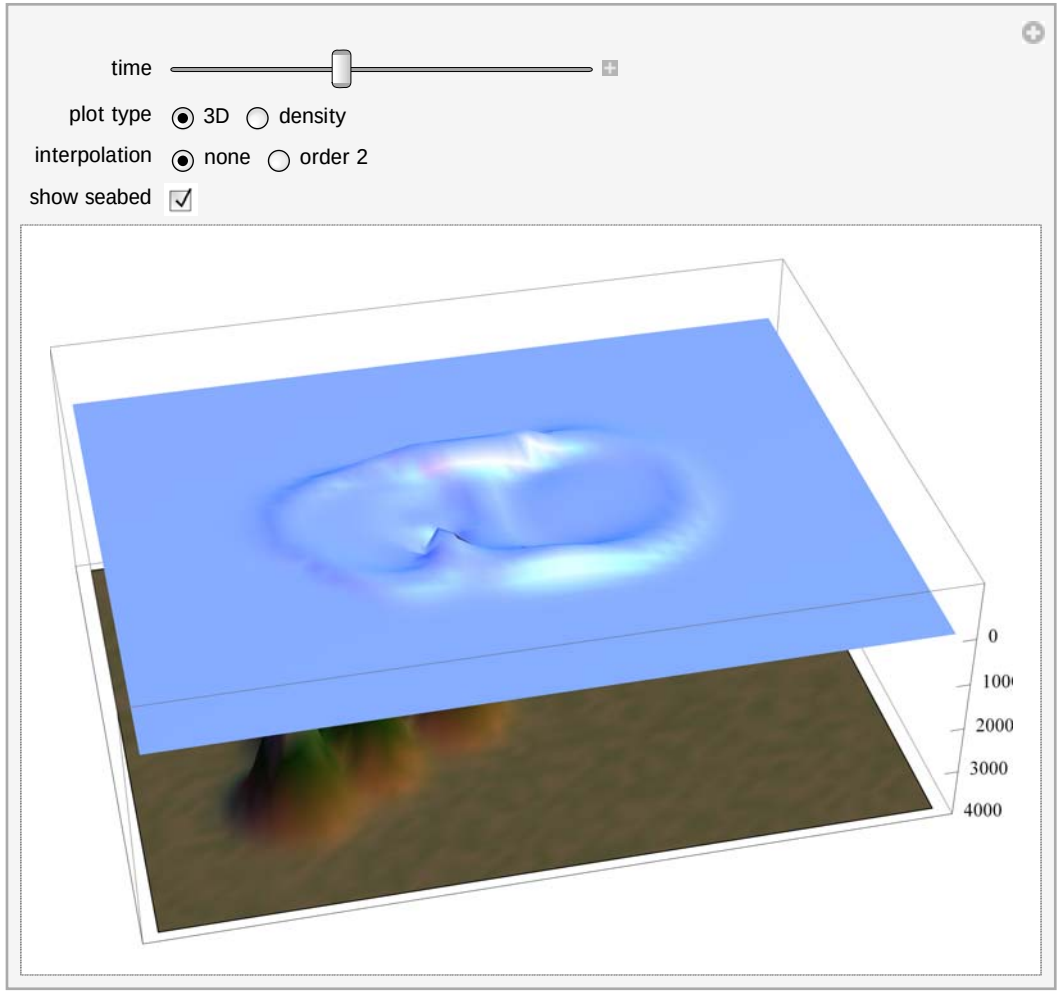
BuildLegend2	PrescribedFilenames	**TScucalcKenterpunkte**	**TSMovingWindow**
CalcOptimalZeroes	**PTSynthese**	**TScucalcVMax**	TSMultiToArray
CalcPhi	ratiotodb	**TScuDetectMainDirectior**	TSPosition
dbtofactor	**ReadBoe**	**TScuNormalize**	TSrescale
Doodson2Frequency	RefPhase	**TSDetermineHole**	**TSScheitel3**
DTenc	**sFFTPTScan**	**TSDetrend**	**TSScheitelFull**
ElementInfo	**sFFTPTScanCluster**	**TSencode**	**TSSelect**
ExportGraphics	tLabel	**TSEreignisKopplung**	**TSStatistik**
ExtractName	TSAggregateLeap	TSExpand	TSthinner
FFTPT	**TScalcMean**	**TSFindFirstExtrema**	**WFBlackman**
FFTPTScan	**TScalcMeanFull**	**TSFrame**	**WFBlackman2**
FindNextTo	**TScalcSpektrum**	**TSHeal**	**WFFejer**
FitPT	**TScalcTidehalbwasser**	**TSHT**	**WFFlatTop**
InitPTList	**TScalcTidehub**	**TSInfo**	**WFGauss**
InitPTListQ	**TScalcTimeshift**	**TSInterpol**	**WFHamming**
JoinKeyTables	**TScalcTMW**	**TSListPlotArray**	**WFHanning**
ListPlotSpektrum	**TSCheck**	**TSListPlotArrayl**	**WFKaiser**
ListPlotSpektruml	**TSCheckAequidistanz**	**TSLowPassFilter**	**WFRechteck**
PegelQ	TSCollapse	TSMakeRaster	**WFTriplett**
PickEverynth	TSCreateFN	**TSMarkLeapinTime**	**WriteBoe**

vorgestellte Befehle

Mathematics of Tsunamis

```
Manipulate[If[plottype == "3D",

  Show[ListPlot3D[Table[sol[λ, ϕ, tt], {λ, λmin, λmax, (λmax - λmin)/nxy}, {ϕ, ϕmin, ϕmax, (ϕmax - ϕmin)/nxy}],
    DataRange → {{ϕmin, ϕmax}, {λmin, λmax}}, PlotRange → {-4, 1.2}, PlotStyle → Directive[RGBColor[.8, .8, 1], Specularity[White, 20]],
    MaxPlotPoints → ∞, InterpolationOrder → io, Mesh → None, PerformanceGoal → "Speed"],
   If[sb, seabed, {}], Lighting → Automatic, BoxRatios → {ϕmax - ϕmin, λmax - λmin, 1/4}, ImageSize → {450, 325},
   Ticks → {None, None, {{-4, 4000}, {-3, 3000}, {-2, 2000}, {-1, 1000}, {0, 0}}}],

  Show[ListDensityPlot[Table[sol[λ, ϕ, tt], {λ, λmin, λmax, (λmax - λmin)/nxy}, {ϕ, ϕmin, ϕmax, (ϕmax - ϕmin)/nxy}],
    DataRange → {{ϕmin, ϕmax}, {λmin, λmax}}, ColorFunction → (ColorData["TemperatureMap"][.8 (#1 + .5)] &),
    ColorFunctionScaling → False, PlotRange → {-1, 1.2}, MaxPlotPoints → ∞, InterpolationOrder → io, Mesh → None],
   If[sb, seabedcontour, {}], AspectRatio → Automatic, ImageSize → {450, 325}, FrameTicks → None]],

{{tt, 0, "time"}, 0, 2 × 3600, 360}, {{plottype, "3D", "plot type"}, {"3D", "density"}, ControlType → RadioButton},
{{io, None, "interpolation"}, {None → "none", 2 → "order 2"}, ControlType → RadioButton},
{{sb, True, "show seabed"}, {True, False}}, AutorunSequencing → {1, 2}, SynchronousInitialization → False,
Initialization :> (

  ldist[p1_, p2_][p_?(VectorQ[#1, NumberQ] &)] := If[p1 == p2, Norm[p - p1], Module[{pp1, pp2, p12}, pp1 = p - p1; pp2 = p - p2;
     p12 = (p1 - p2)/Norm[p1 - p2]; If[Sign[p12.pp1] ≠ Sign[p12.pp2], Norm[pp1 - pp1.p12 p12], Min[Norm[pp1], Norm[pp2]]]]];
```

```
seamount[s_, w_, {λ_, ϕ_}][l_, p_] := s ℯ^(-w ((l-λ)^2+(p-ϕ)^2));
λmin = -15 °; λmax = 15 °; λe = 0; λi = 2.5 °; ϕmin = -35 °; ϕmax = 5 °; ϕ1 = -10 °; ϕ2 = -20 °; ϕi = -25 °; T = 2 × 3600; nxy = 50;

b[λ_, ϕ_] := -4000 + seamount[3500, 1000, {-2.5 °, -25 °}][λ, ϕ] + seamount[3000, 1000, {-5 °, -23 °}][λ, ϕ] +
  seamount[3000, 1000, {-7.5 °, -27 °}][λ, ϕ] + seamount[3000, 1000, {5 °, -24 °}][λ, ϕ] +
  seamount[3000, 1000, {5 °, -21 °}][λ, ϕ] + seamount[3000, 1000, {5 °, -18 °}][λ, ϕ] + seamount[3000, 1000, {5 °, -15 °}][λ, ϕ] +
  seamount[3000, 1000, {5 °, -12 °}][λ, ϕ] + seamount[3750, 1000, {-2 °, -18 °}][λ, ϕ];

SeedRandom[2004];

seabed = Plot3D[.001 b[u, v], {v, ϕmin, ϕmax}, {u, λmin, λmax}, PlotRange → All, PlotPoints → nxy, Mesh → None,
  MaxRecursion → 0, ColorFunction → (Blend[{Brown, RGBColor[0, 0.25, 0], White}, #3 + RandomReal[.1]] &)];
seabedcontour = ContourPlot[b[u, v], {v, ϕmin, ϕmax}, {u, λmin, λmax}, PlotRange → All, PlotPoints → nxy, Mesh → None, MaxRecursion → 1,
  ContourShading → None, ContourStyle → Opacity[.15], Contours → {-3500, -3000, -2500, -2000, -1500, -1000, -500}];

swe = {∂t h[λ, ϕ, t] + 1/(R Cos[ϕ]) (∂λ (u[λ, ϕ, t] (h[λ, ϕ, t] - b[λ, ϕ])) + ∂ϕ (v[λ, ϕ, t] (h[λ, ϕ, t] - b[λ, ϕ]) Cos[ϕ])) == 0,
  ∂ϕ u[λ, ϕ, t] v[λ, ϕ, t]/R + g ∂λ h[λ, ϕ, t]/(R Cos[ϕ]) + ∂t u[λ, ϕ, t] + u[λ, ϕ, t] ∂λ u[λ, ϕ, t]/(R Cos[ϕ]) == f v[λ, ϕ, t],
  ∂ϕ v[λ, ϕ, t] v[λ, ϕ, t]/R + g ∂ϕ h[λ, ϕ, t]/R + ∂t v[λ, ϕ, t] + u[λ, ϕ, t] ∂λ v[λ, ϕ, t]/(R Cos[ϕ]) == -f u[λ, ϕ, t]};

Block[{R = 6 378 000, ω = 2 π/(24 × 3600), f = 2 ω Sin[ϕ], g = 9.8, s = .95},
 sol = First[h /. NDSolve[{swe, h[λ, ϕ, 0] == ℯ^(-2000 ldist[{λe, ϕ1}, {λe, ϕ2}][{λ, ϕ}]^2), u[λ, ϕ, 0] == 0, v[λ, ϕ, 0] == 0,
      h[λ, ϕmin, t] == h[λ, ϕmax, t], h[λmin, ϕ, t] == h[λmax, ϕ, t]}, h, {λ, λmin, λmax}, {ϕ, ϕmin, ϕmax},
     {t, 0, T}, DependentVariables → {h, u, v}, Method → {"MethodOfLines", "SpatialDiscretization" →
        {"TensorProductGrid", "MinPoints" → {nxy, nxy}, "MaxPoints" → {nxy, nxy}, "DifferenceOrder" → "Pseudospectral"}}]]];
)]
```



THIS NOTEBOOK IS THE SOURCE CODE FROM

"Mathematics of Tsunamis" from The Wolfram Demonstrations Project
http://demonstrations.wolfram.com/MathematicsOfTsunamis/

- Contributed by: Yu-Sung Chang
- After work by: Rob Knapp and Roger Germundsson

Interesse?

Was wird benötigt:

- *Matematica 7*
- *Datei MKToolMM7_log.dat* → *Zentraler Ablageort?*

Interesse an einem Vortrag/Workshop Spektral-/Partialtidenanalyse?