Scilab-Programmkomplex zur Planckschen Strahlungsgleichung

- Funktionsüberblick

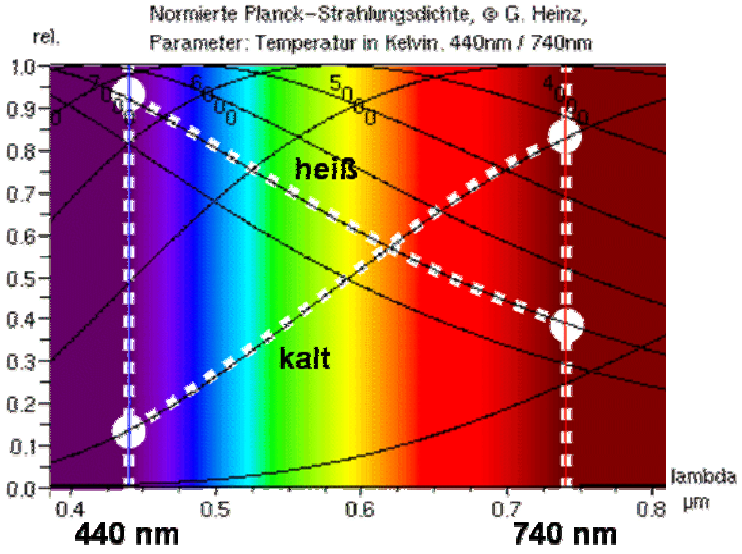
- Programme
  - tempfunktion_entwickeln.sce
  - tempfunktion_plotten.sce
  - wav2gif_v6.sce

- Freeware Scilab 3.1.1
  - Scilab
    http://www.scilab.org/
  - Downloads
    http://www.scilab.org/communities/developer_zone/scilab_versions/oldreleases/scilab_3.1.1

Impressum
Dr. G. Heinz, GFaI
Vollmerstr.3
12489 Berlin
www.gfai.de/~heinz
heinz@gfai.de
Mob. 0172 545 6036

Funktionsüberblick

- Plancksche Schwarzkörperstrahlung wird vorausgesetzt (i.a. > 1µm)
- Zwei gemessene Emissionswerte *e1*(*λ1*), *e2*(*λ2*) definieren eine Temperatur T über die Planksche Strahlungsgleichung, z.B. *λ1* = 440nm, *λ2* = 740nm siehe Bild
- Dann kann man schreiben:
  e1 = e1(T) und *e2 = e2(T)* mit *T = T*
- Umgekehrt wird jede Temperaturkurve mit diesen zwei Punkten *e1, e2* definiert:
  T = T(*e1,e2*)
- Das Programm
  tempfunktion_entwickeln.sce
  generiert eine Temperaturfunktion zweier Lambda als **.bin* - Tabelle mit *T = T*(*e1/e2*)
  (einmalig für jede Diodenpaarung zu tun)
- Dieser wird gesteuert durch **.pin* (editierbarer ASCII-File)



Funktionsüberblick

- Die für zwei Lambda aus der Planckschen Strahlungsgleichung gewonnene Temperaturfunktion wird gespeichert als **.bin* in einer Matrix 'vierer' aus *4* Spalten und *n* Zeilen (*n* Samplezahl)
- Matrix 'vierer':
  - 1.Spalte: Temperatur in Kelvin
  - 2.Spalte: Emissionswerte *e1*
  - 3.Spalte: Emissionswerte *e2*
  - 4.Spalte: Quotient *e1/e2*
- Beispiel: Test auf Scilab-Konsole --> size(vierer)
- Werte aus Zeitfunktionen (*.wav) greifen über lineare Interpolation in diesen File, um die zugehörige Temperatur zu erhalten
- Zur Berechnung der Zeitfunktionen wird eine Quotientenform genutzt
- Der Quotient der Emissionswerte (*e1/e2*) entspricht einer Temperatur *T*
  T = T(e1/e2)
- Mit *tempfunktion_plotten.sce*
  kann eine vorhandene Temperaturfunktion eingelesen und geplottet werden, wichtig zur Prüfung der Funktion auf Monotonie(!)

tempfunktion_entwickeln.sce

Input:

*.pin Textfile zur Parametereinstellung –
wird bei Start im Dialog abgefragt

Output:

*.bin Binärfile mit der Temperaturfunktion als
Vierermatrix vierer=[grad,e1,e2,e1zue2]

- » grad: Temperatur in Kelvin (Output)
- » e1, e2: Emissionswerte (Input) (unbenutzt)
- » e1zue2: Ratio e1 zu e2 als Input

Scilab-Graphic(0) bis (3): Sichten auf die Temp.-Funktion, b.w.

*.gif Bild der entwickelten Temperaturfunktion (Fenster #3)

tempfunktion_entwickeln.sce – Parameterinit *.pin

Beispiel tempfkt_1550nm_2300nm.pin

```
//  Parameter-Init für Planck-Temperaturkurve entwickeln

// Bestimmungs-Wellenlängen:
      L1=1.55;     // Empfindlichkeitsmaximum erster Photodiode in µm
      L2=2.3;      // Empfindlichkeitsmaximum zweiter Photodiode in µm
      Ln=300;      // Anzahl der lambda-Stützpunkte pro Kurve
      Lstart=1.0;  // Von Wellenlänge [in µm]
      Lstop=3.0;   // Bis Wellenlänge [in µm]
      m=0.000001;  // Korrekturfaktor, wenn lambda in Mikrometer eingesetzt wird

// Temperaturen:
      Tanf=300;    // Beginnen bei Temperatur
      Tend=3000;   // Endtemperatur
      T=Tanf;      // Temperatur in Kelvin
      dT = 10     // Temperatur-Schrittweite für hübsches Bild
      //dT= (Tend-Tanf)/Ln ;     // Temperatur-Schrittweite für Ablage in *.bin-File
      tempk = 1;   // Absolut-Kalibrierung der Temperatur

// Strings:
      filename='tempfkt_'+string(L1*1e3)+'nm'+'_'+string(L2*1e3)+'nm'; // für Ausgabefiles
      col='black'; // Farbe des Ergebnisplots #3
```

tempfunktion_entwickeln.sce - Scilab-Dialog

```
Approximation der Planck-Temperatur
aus dem Pegel zweier Spektrallinien

Autor: G. Heinz, GFaI-Berlin, 12/2007 - 1/2008
heinz@gfai.de, www.gfai.de/~heinz
Copyrights: Uneingeschränkte Nutzung erlaubt bei
        Quellenverweis.

Input:  Emissionspegel e1 and e2 bei Wellenlänge L1 and
        L2 in µm
Output: Temperaturfunktion T = f(e1/e2)

Absolutpegel von e1 und e2 sind unbekannt ->
        Kalibrierung mit tempk
 dT  =    100.

From parameter file:
C:\Daten\_Bibliothek\Bits_and_Bytes\Scilab\wav2gif_TU\
        tempfkt_1950nm_2300nm.pin

L1 = 1.950000 µm
L2 = 2.300000 µm
dT = 100.000000 K

... GIF-Plot of Window 0 saved as
        tempfkt_1950nm_2300nm_0_planck.gif

Achtung: Variablenname vierer wird mit gespeichert

Temperature function saved as
wrfile = tempfkt_1950nm_2300nm_27x4.bin
with matrix-format vierer=[grad,e1,e2,e1zue2]

... GIF-Plot of Window 1 saved as
tempfkt_1950nm_2300nm_1.gif
... GIF-Plot of Window 2 saved as
tempfkt_1950nm_2300nm_2_r-b.gif
... GIF-Plot of Window 3 saved as
tempfkt_1950nm_2300nm_3_quot.gif

Temperature function saved under
wrfile = tempfkt_1950nm_2300nm_27x4.bin
with matrix-format vierer=[grad,e1,e2,e1zue2]

Name                Type         Size         Bytes
vierer              constant     27 by 4      880

for test:
-->size(vierer)
-->vierer
-->whos -name vierer

Example for reload:
-->wrfile
-->clear 'vierer'
-->load(wrfile)
-->vierer

End.

GIF-Files saved at
C:\Daten\_Bibliothek\Bits_and_Bytes\Scilab\wav2gif_TU
```

tempfunktion_entwickeln.sce – Scilab-Windows

Scilab-Graphic(0…3):

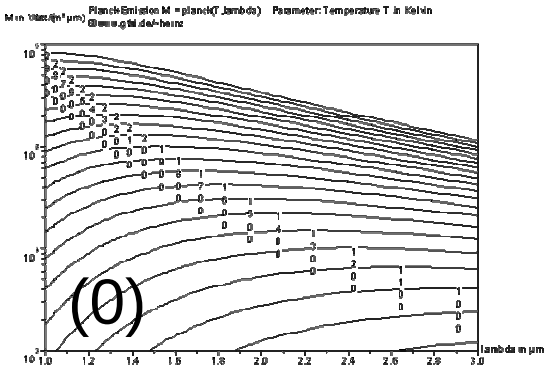
(0): Plancksche Schwarzkörperstrahlung

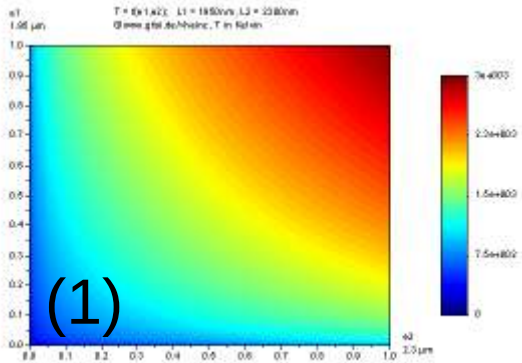
(1): 2d-Plot T(e1,e2)

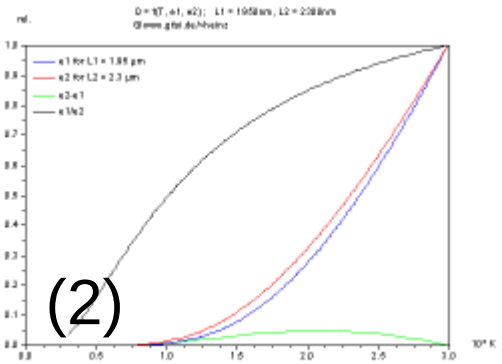
(2): Test-Funktionen

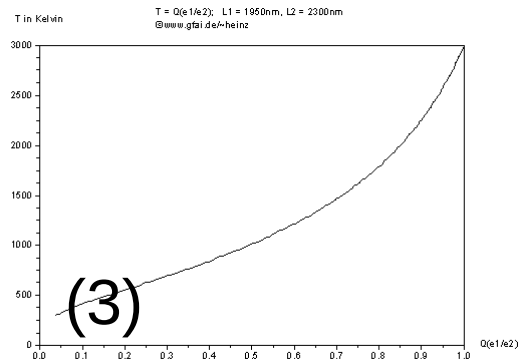
(3): Ausgabe-File

T = T(e1/e2)









tempfunktion_plotten.sce

Input: *.bin Temperaturfunktion

Output: *.gif

- Die Funktion dient der Prüfung der Temperaturfunktion auf Monotonie - nur dann ist diese nicht mehrdeutig
- Die Funktion kann auch zum Einlesen der Temperaturfunktion genutzt werden um Zugriff auf die Variablen zu erhalten
- Einmal starten, dann z.B. auf Scilab-Kommandozeile eingeben:
  - → who
  - → wrfile
  - → vierer
  - → vierer(1,:) 1.Reihe
  - → vierer(2,:) 2.Reihe
  - → vierer(:,1) 1.Spalte
  - → vierer(:,2) 2.Spalte
  - → …

tempfunktion_plotten.sce – Scilab-Dialog

Temperaturfunktion *.bin plotten
heinz@gfai.de

vierer = Matrix aus vier Spalten

Current working directory:
C:\Daten_Bibliothek\Bits_and_Bytes\Scilab\wav2gif_TU

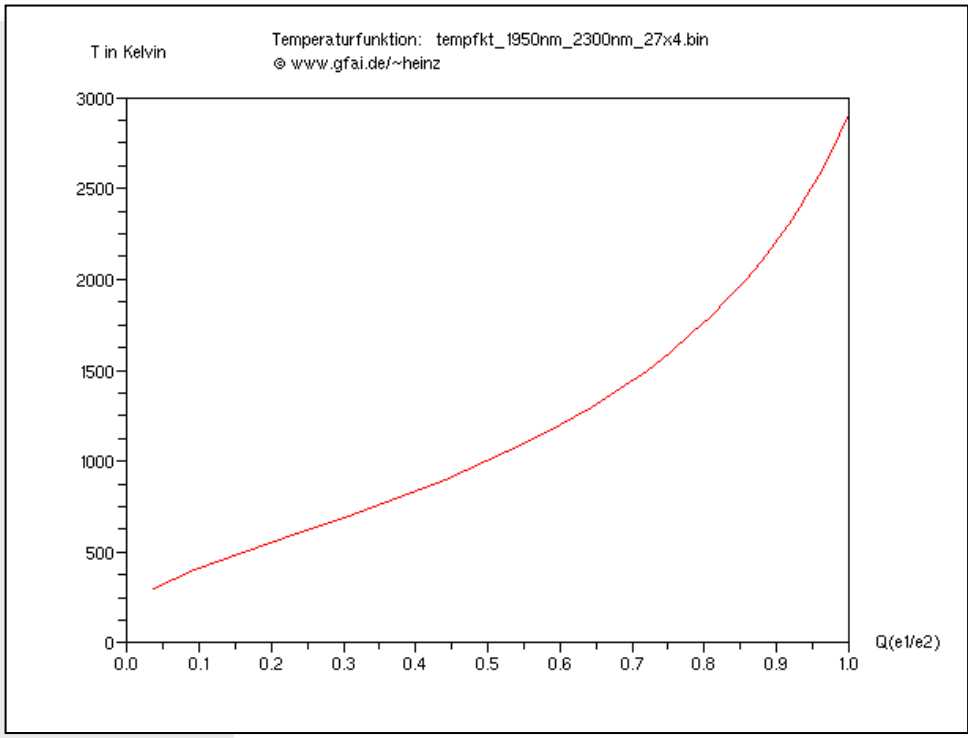
Input-File:
tempfkt_1950nm_2300nm_27x4.bin

Read-size: vierer in 27 Reihen mit 4 Spalten

vierer:
 1.Spalte: Temperatur in Kelvin
 2.Spalte: Werte e1
 3.Spalte: Werte e2
 4.Spalte: Quotient e1/e2

String-Matrix mit 27x4 Werten wurde als vierer(grad,e1,e2,ratio) gelesen

Tests:
-->vierer
-->size(vierer)
-->typeof(vierer)



wav2gif_v6.sce – Wavs als Gifs plotten

Funktion:

- Plot von max. 8 Wav-Zeitfunktionen f(…) und
- Plot von max. 8 aus Wavs berechneten Zeitfunktionen g(…)

Input:

*.wav	zu plottende Zeitfunktionen
*.pra	Parameter-Initialisierung - in diesem File werden alle Funktionen definiert und editiert. Er wird im Dialog abgefragt.

Output:

*.gif	Plotbild der Zeitfunktionen
*.csv	Excel-Tabelle der Zeitfunktionen
*.inc	AVR-Studio4 Wertetabelle für Linker (Atmel-Prozessoren)

wav2gif_v6.sce – Parameterfile *.pra

```
//
// Steuerfile für WAV2GIF.SCE v6
//
// Beispiel zur Namensgebung:
// Ursprung:            mist.chl                (source: mist)
// abgeleitet:          mist_IR.wav, mist_UV.wav, mist_blau.wav etc.
// "blau" ist z.B. als parameter_1 "filename" referenziert
//
parameter='relativ';
source='A664-0398'; // chl-Stammname hier eintragen
//
// Plot-Überschrift
titel='Schweißversuch mit Photodioden';
//
// Plotbereich der wav-Files
tstart=106;                             // Start in Millisekunden
//start=4680;                           // Start in Samples
//tdelta=10;                            // Dauer in Millisekunden
delta=256;              // Dauer in Samples
//
// Anzeigebereich Plotbild
yo=-.1; // Anfang Wertebereich y
yr=1.2; // Ende Wertebereich y
//
// Labels - Legendenplatzierung
xL=200;                 // Label-Startposition Sample x
yL=.3;  // Label-Startposition Sample y
yk=.1;                  // Labelabstand y
//
// Wertebereich xo,xr wird durch delta und fs erzeugt
// Samplerate fs kommt aus *.wav
//
// WAVs plotten:
// Parameter: 1_wavname 2_farbe 3_funktion 4_funktion 5_plot?
// für fx stehen alle Funktionen der Form f(y) und f(y,const) bereit
// erster Name muß mit *.wav -Kürzel identisch sein
//
anzf=6; // Anzahl zu plottender WAVs, max.8, bitte durchgehend nummerieren
f1='Isch black Amp(y,.3) shift(y,.05) yes';  // Amp für I-Kalibrierung
f2='Usch gray Volt(y,1) shift(y,.05) yes';    // Volt für U-Kalibrierung
f3='440nm purple scale(y,1) norma(y) es'; //
f4='740nm brown scale(y,1) norma(y) es'; //
f5='1550nm blue scale(y,1) norma(y) yes';                       //
//f6='1950nm yellow scale(y,1) norma(y) yes';                   //
f6='2300nm red scale(y,1) norma(y) yes';                        //
//
//
// Funktionen berechnen:
// Parameter: 1_funktion 2_farbe 3_skalierung 4_plot?
// für Parameter 1, 3 und 4 stehen alle Funktionen bereit
// Zugriff auf f1...f6 möglich
//
anzg=2;
                                        // Anzahl zu berechnender Zeitfkt., max.8
g1='TB(f1,f2,4) orange scale(y,1e-3) yes';              // Stefan/Boltzmann Temperatur
//
g2='TP(f5,f6,1) green scale(y,2e-4) yes';               // Planck-Temperatur (blau, rot)
// Planck-Temperaturkurve laden:
kurve='tempfkt_1550nm_2300nm_27x4.bin';                                 // File muß
        vorhanden sein!
//
//g3='P(f3,f4) orange scale(y,1e-4) yes';               // Power (scale in kW~10e-3, W~1)
//g4='E(f3,f4) orange scale(y,1) yes';                  // zugef. Energy in mW pro Sample
//g5='S(f1,f2,0,-.1,1,0) purple scale(y,1) -';          // Schwellwert
//g6='f1-f2 yellow scale(y,1) norma(y) -';  // Subtraktion
//g7='f6-f5 green scale(y,1) yes';                                      // Differenz
//g1='TP(f5,f6,1) green norma(y) yes';                  // Planck-Temperatur
//g2='TB(f5,f6,10) orange norma(y) yes';                // Boltzmann-Temperatur
//g1='TP(f5,f6,1) green scale(y,1e-4) yes';             // Planck-Temperatur
//g2='TB(f5,f6,10) orange scale(y,2e-4) yes';           // Boltzmann-Temperatur
//
// Zeitfunktionen zur Ausgabe an *.inc File Atmel AVR-Studio4, Länge 256 mal 4
avrfunc='f3 f4 f5 f6';                  // genau vier f-Funktionen sind erlaubt
avrrate = 10000;      // 10kHz ~ 1/100 µs, gewünschte Sampelrate für inc-File
avrleng = 256;                                          // gewünschte Anzahl von Sampels
//
```

wav2gif_v6.sce – Scilab-Dialog

- Dialogführung siehe Scilab-Dialog.pdf

Ergebnisplot: